


# WinSyslog Configuration Documentation

## version 26.11

**Adiscon GmbH**

November 04, 2026

# Contents


**Manual** **2**

- Getting Started 2
  - Installation 2
  - Understand the Components 2
  - Receive Logs 3
  - Creating an Initial Configuration 4
  - Process and Filter 5
  - Store and Forward 6
  - Operate and Troubleshoot 6
- Tutorials 7
  - Tutorial: Create a Simple Syslog Server and Write to a File 7
  - Quick Start: Write Syslog Messages to Microsoft SQL Server 9
  - Tutorial: Integrate WinSyslog with a Custom Database Schema 14
  - Tutorial: Prepare WinSyslog Data for Adiscon LogAnalyzer 17
  - Tutorial: Export the Configuration and Create a Debug Log 18
  - Tutorial: Configure a SETP Server Service 19
  - Tutorial: Forward Messages to Another Syslog Server 20
  - Tutorial: Send Matching Messages by Email 21
- InterActive SyslogViewer 21
  - InterActive SyslogViewer 22
  - Options & Configuration 22
    - Launching InterActive SyslogViewer 23
    - Using InterActive SyslogViewer 24
    - Options & Menus 25
    - Live Syslog View 32
    - Database View 34
  - FAQ 35
    - How to Autostart Interactive Syslog Viewer 36
- Configuring 38
  - Core concepts 38
    - WinSyslog - Services 39
    - Information Units 40
    - Rules 41
    - The Rule Engine 44
    - Filter Conditions 52
    - Actions 54
  - Configuring WinSyslog 55
  - Organizing with RuleSets, Rules, and Actions 57
  - Client Options 58
  - Client Tools 60

Using File based configuration 66
General Options 71
License 72
General 73
Debug 77
Engine 79
Queue Manager 84
Permitted Senders 86
Post Process Apps 87
Services 88
Heartbeat 89
MonitorWare Echo Reply 91
RELP Listener 92
SETP Server 95
SNMP Trap Receiver 98
Syslog server 101
Filter conditions 110
Global Conditions 113
Date Conditions 114
Operators 115
Filters 116
General 117
Date/Time 119
InformationUnit Type 120
Syslog 121
SNMP Traps 122
Event Log Monitor 123
Event Log Monitor V2 125
File Monitor 127
Custom Property 128
Extended Number Property 129
Extended IP Property 130
File Exists 132
Store Filter Results 133
Actions 133
ODBC Database Options 134
OLEDB Database Action 142
File Logging Options 148
Event Log Options 158
Send Email 160
HTTP REST Output 166
Net Send 170

Send to Communications Port 171
Send MSQueue 174
Send RELP 175
Send SETP 180
Send SNMP Trap 184
Syslog Forwarding 190
Send DTLS 203
HTTP Request 205
Call RuleSet 207
Compute Status Variable 208
Discard 209
Normalize Event 210
Post Processing 211
Parsing log messages 213
Resolve Hostname Action 221
Message Ringbuffer Action 222
Set Property 224
Set Status 225
Play Sound 226
Start Program 227
FAQ 228
Why were dynamic log files sometimes not rotated in older WinSyslog versions? 228
Why were dynamic log files sometimes not rotated in older WinSyslog versions? 229
Log Rotation Naming Convention Change in WinSyslog 18.x 229
Why does log rotation fail when using ZIP compression in WinSyslog? 232
Are WinSyslog products affected by recent OpenSSL CVEs? 233
Troubleshooting the Start Program action in WinSyslog 233
High-load configuration reload issues in WinSyslog 235
Queue Buildup During SQL Server Table Cleanup Operations in WinSyslog 236
Default Timevalues Setting in WinSyslog Explained 239
How to Export WinSyslog Settings for a Support Call 241
Recommended Service Stop Order for WinSyslog Maintenance 241
Running WinSyslog on a Windows Cluster Server 243
Why Does My WinSyslog Database Setup Fail? 244
Why Does a WinSyslog Message Show Two Timestamps? 246
What Does WinSyslog Event ID 1011 Mean? 246
Why did my WinSyslog update report success when the files were not upgraded correctly? 247
Why do log files remain locked when multiple rules write to the same file? 254
How to resolve performance issues on high-load systems? 256
Is MariaDB supported by the ODBC action? 259
Recommended Palo Alto Firewall Syslog Configuration 260
How Do I Perform a Repeatable Deployment of WinSyslog? 262

Why did my WinSyslog update report success when the files were not upgraded correctly? 265
How to Copy the WinSyslog Configuration to Other Servers 272
How Do Remote Administration and Browser-Based Log Review Fit Together? 273
How Can I Obtain a Printable Version of the WinSyslog Manual? 274
Can WinSyslog Write to a UNC Path? 274
Which Database Format Should I Use with WinSyslog? 275
What do “service”, “input”, “listener”, and “server” mean in WinSyslog? 276
How Do Port, Address, and Transport Conflicts Work for Input Services? 277
What do CA PEM, Certificate PEM, and Key PEM mean for TLS input services? 279
Forwarding syslog messages with original IP address in WinSyslog 279
What Is the Difference Between SETP and Syslog? 281
Why can WinSyslog use EventReporter properties? 282
How Do I Handle Non-Standard Syslog Messages from Unix or Linux Systems? 283
How Do I Enter WinSyslog License Information? 284
What is Freeware Mode? 284
WinSyslog vs Kiwi Syslog Server – Which to Choose? 285
Why did product versions jump to 26.07? 287
What happens when the client opens a configuration with unsupported blocks? 288
Why is Net Send no longer available in the configuration client? 288
How do I set up HTTP REST Output? 289
Do the configuration clients require .NET Framework, or is .NET Core or .NET 5+ enough? 290
Is WinSyslog v18+ supported on Windows Server IoT 2025? 291
Sales 291
How do I contact Adiscon sales? 292
What should I include in a quote request? 292
What happens after I open a sales ticket? 293
How do purchase orders and billing requests work? 294
Licensing and ordering 295
Can I use an older product version with my license? 296
Air-gapped environments 297
Offline installation and activation 297
Online verification after activation 298
Perpetual licenses and UpgradeInsurance 299
UpgradeInsurance 299
Reference 300
WinSyslog Shortcut Keys 300
Command Line Switches 301
Edition Comparison 301
Comparison of properties 302
Event Properties 302
Accessing Properties 303
System Properties 308

Custom Properties 309
Event-Specific Properties 310
Complex Filter Conditions 317
Connect to Computer 320
Registry Paths 321
System Error Codes 321
User Interface changes 321
Glossary 323
Engine Only Install 324
FTP 325
HTTP 326
IETF 327
IMAP 328
IPv6 329
Millisecond 330
NNTP 331
POP3 332
Registry File 333
RELP 334
Resource ID 335
RFC 3164 336
RFC 3195 337
RFC 5424 338
SETP 339
SMTP 340
SNMP 341
Syslog Facility 342
TCP 343
UDP 344
UTC 345
WinSyslog product Event ID reference 345
WinSyslog Event ID troubleshooting procedures 346
Known issues and errata 354
Copyrights 354

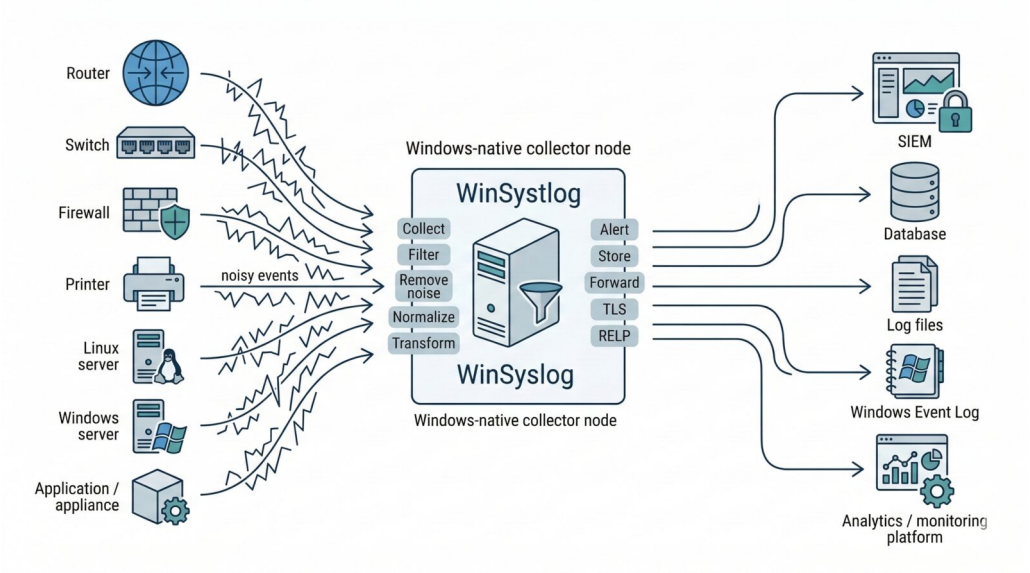
# About WinSyslog

**WinSyslog is a Windows syslog server for collecting, processing, storing, and forwarding log messages across Windows-centric and mixed environments. It helps IT operations, security, and compliance teams centralize syslog data from network devices, appliances, servers, and applications on a native Windows platform.**

Developed since 1996, WinSyslog was the first syslog server for Windows. It is built by the same team behind rsyslog and combines long-standing syslog expertise with a Windows-focused deployment and management model. Adiscon has published additional background on this shared engineering lineage in The rsyslog Evolution: Bridging BSD Heritage with Adiscon Innovation.

Syslog remains one of the standard protocols for centralized event logging. Routers, switches, firewalls, printers, hypervisors, Linux systems, and many security appliances can all send syslog messages. WinSyslog provides the Windows-side collection point for these events and lets you route them to the destinations and workflows your environment requires.

WinSyslog is designed for environments that need more than a basic message receiver. It can receive syslog over UDP and TCP, support secure syslog transport with TLS, and integrate with RELP-based forwarding for reliable log delivery. It also runs as a native Windows service and fits naturally into Windows administration and monitoring practices.




*WinSyslog can act as a Windows edge collector that reduces noise and forwards cleaner event streams to SIEMs, databases, files, and other downstream systems.*

Typical use cases include:

- centralized logging for routers, switches, firewalls, wireless controllers, printers, and other syslog-capable devices
- real-time alerting and event-driven automation for critical infrastructure and security incidents
- long-term log storage in text files, ODBC databases, and the Windows Event Log
- secure forwarding and relay scenarios that require TCP, TLS, or reliable log transport
- edge or node-level collection in Windows environments, where WinSyslog can remove noise, normalize messages, and forward cleaner event streams upstream
- Windows-based syslog collection for SIEM pre-filtering, compliance retention, and operational troubleshooting

WinSyslog uses a flexible rule engine to filter, enrich, store, display, and forward events. You can use it as a standalone Windows syslog server or as part of a larger logging architecture with other Adiscon components and downstream analysis systems. In distributed logging pipelines, this also helps reduce unnecessary upstream traffic, storage volume, and processing load by dropping low-value events early and forwarding only the data that needs further retention or analysis.

This manual covers installation, features, configuration, operations, FAQ content, and reference material for WinSyslog. Use the sections below to get started or go directly to the area you need.

# Manual

## Getting Started

Start here to understand what WinSyslog is, how its main components work together, how logs flow through it, and how to build a first working configuration.

### Installation

**orphan:**

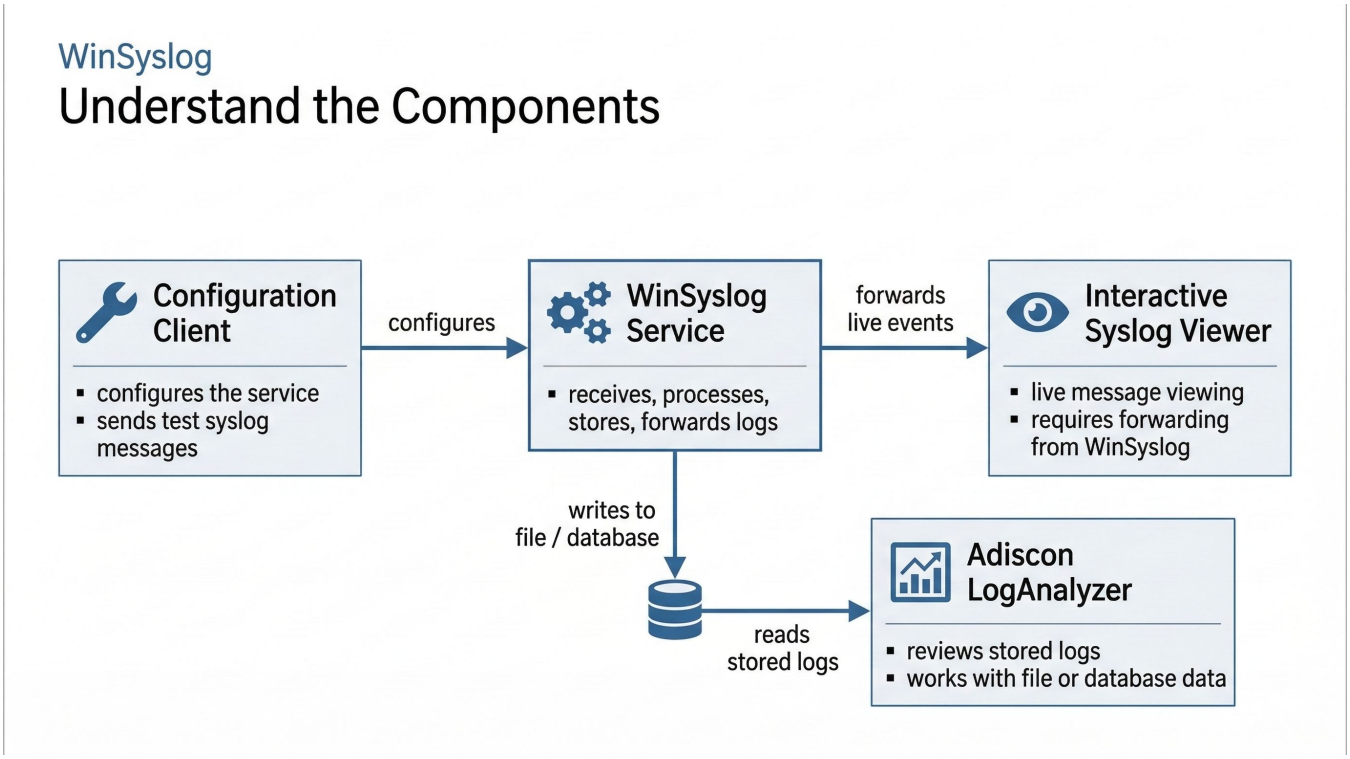
Installing the product is straightforward because a familiar Windows setup program guides you through the process.

Multiple download variants are available so you can pick the package that matches your environment; always install the most recent release to benefit from the latest fixes and improvements.

Each installer automatically creates a Windows Firewall exception for the service process during setup, ensuring the service can communicate right away without additional manual steps.

### Understand the Components

WinSyslog is easiest to understand if you separate its main components by role.




*WinSyslog consists of a background service, a configuration client, a live viewer, and an optional analysis component for stored logs.*

#### The four components

1. **WinSyslog Service** This is the main runtime component. It runs in the background as a Windows service, receives log data through configured input services, processes it through rulesets and actions, and stores or forwards the results.
2. **WinSyslog Configuration Client** This is the administrative tool. You use it to configure input services, rulesets, filters, and actions. Changes are made in the Configuration Client and then applied to the WinSyslog service. It is also where you send a test syslog message from the `Tools` menu.

3. **Interactive Syslog Viewer** This is the live-view component for interactive monitoring. It does not show data automatically by itself. WinSyslog must forward messages to it via a rule.
4. **Adiscon LogAnalyzer** This is an optional analysis component for working with stored logs, especially in files or databases. It is for browsing and reviewing retained data, not for live message intake. Users who prefer a simpler standalone tool for direct log file review may also consider the third-party log viewer Retrospective by centeractive.

### How they fit together

- The **WinSyslog service** runs the configured input services, rulesets, and actions that do the actual receiving, processing, storing, and forwarding.
- The **configuration client** defines what those input services, rulesets, and actions should do and is where configuration changes are applied before the runtime service uses them.
- The **Interactive Syslog Viewer** can display incoming messages live if the WinSyslog service forwards them there.
- **LogAnalyzer** works on stored log data after the WinSyslog service has written it to a file or database.

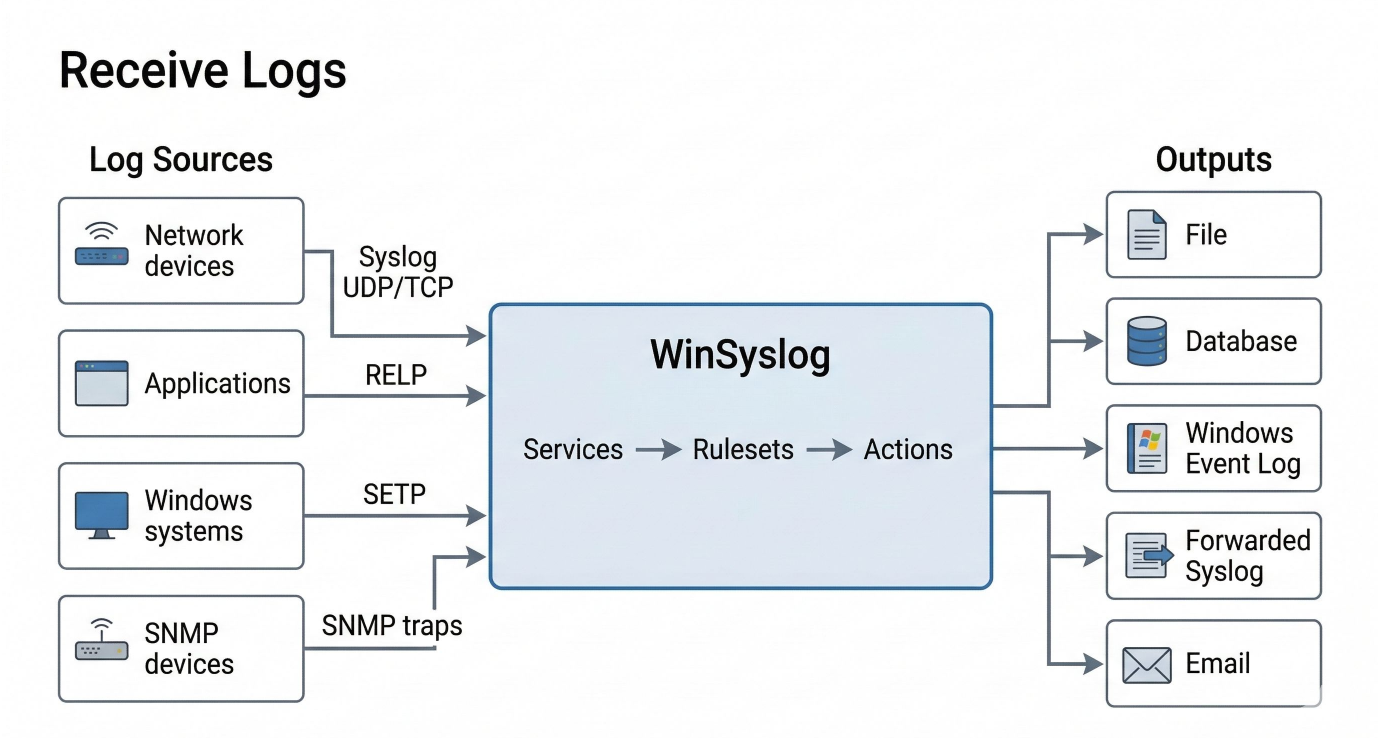
For the current split between remote administration and browser-based review, see How Do Remote Administration and Browser-Based Log Review Fit Together?.

### What to read next

- To understand message intake, start with Receive logs.
- To understand what the service does after receiving data, continue with Process and filter.
- To understand where data can go next, see Store and forward.
- For the more detailed legacy explanation, see `../components` and `../componentsworktogether`.

## Receive Logs

WinSyslog receives logs from multiple sources and turns them into events that can be processed by rules.




*WinSyslog can receive logs from multiple source types, process them through input services, rulesets, and actions, and then store or forward them to downstream targets.*

In this manual, **input** is the clearest plain-language concept for receive configuration, while **service** remains the main operational term. Some GUI pages still use exact labels such as `Syslog server`, `RELP Listener`, and `SETP Server` for specific service types.

What you can receive:

- Syslog over UDP/TCP and secure syslog over TLS
- RELP (reliable transport)
- Windows Event Log events
- SNMP traps

Where to configure it:

- Services provide the configured input services.
- Syslog server service receives syslog.
- RELP Listener service receives RELP.
- SETP Server service receives SETP.
- SNMP Trap Receiver service receives SNMP traps.
- If you run multiple input services, see How Do Port, Address, and Transport Conflicts Work for Input Services? before reusing a port for another service. In that FAQ, `listener` refers to the network side of a service.

Quick verification:

- In the WinSyslog Configuration Client, open *Tools* and use *Send Syslog Test Message* (see **`Send Syslog Test Message`**).
- Confirm messages arrive in the configured ruleset (for example, write to a file).

## Creating an Initial Configuration

Use this page to build the first working WinSyslog configuration: receive a test syslog message and write it to a local file.

### Goal

At the end of this procedure, WinSyslog will:

- listen for incoming syslog messages
- process them through a ruleset
- write them to a local file

For terminology: in this manual, the plain-language concept is **input**, while the configured object is a **service**. The current GUI name for the syslog input service is `Syslog server` service.

### Prerequisites

- WinSyslog is installed.
- You can open the WinSyslog Configuration Client.
- The WinSyslog service is installed on the system.

### Steps

1. Create a ruleset.
    - In the WinSyslog Configuration Client, create a new ruleset.
    - Leave filtering simple for the first test so that all incoming messages can match.
2. Add one file action to that ruleset.
    - Inside the ruleset, add a Write to File action.
    - Choose an easy-to-find test file path.
3. Create one input service to receive syslog.
    - Under Services, add a Syslog server service.
    - Bind that input service to the ruleset you created.

- Keep the default input settings unless you already know you need a different port or protocol.

4. Save the configuration.
   - Apply or save the changes in the Configuration Client so the input service can use them.
   - Until you apply the changes, the running service continues to use the previous configuration.
5. Start or restart the WinSyslog service if required.
   - Ensure the WinSyslog service is running with the new configuration.

### How to verify

1. In the WinSyslog Configuration Client, use `Tools -> Send Syslog Test Message` (see **`Send Syslog Test Message`**).
2. Confirm that the message is written to the file configured in the action.
3. If nothing arrives, check:
   - the `Syslog server` service is enabled
   - the input service is bound to the correct ruleset
   - the file action is inside that ruleset
   - the WinSyslog service is running

### Expected result

If the configuration is correct, WinSyslog receives the test message and stores it in the configured file. At that point, you have a working end-to-end setup.

### Next step

- To understand message intake options, see Receive Logs.
- To add filtering and more advanced processing, continue with Process and Filter.
- To add forwarding or additional storage targets, continue with Store and Forward.

## Process and Filter

WinSyslog uses a rules engine to decide what to do with each incoming event: drop it, store it, forward it, or trigger notifications.

### Where to configure

- Configuring WinSyslog explains the tree view (Services, RuleSets, rules).
- Filter conditions decide which events match a rule.
- Actions define what happens for matching events.

### Recommended setup path

1. Start with one input service under Services and attach it to a ruleset.
2. In the target ruleset, add one rule with a single, simple action (for example, Write to File).
3. Add filter conditions to narrow down the events: - Start broad (facility, severity, source) - Then add message/content filters
4. Add additional actions once the rule matches exactly what you intend.

### Things that commonly trip people up

- Rule order matters: rules are evaluated top-to-bottom inside a ruleset.
- An input service decides which ruleset sees an event. If events “disappear”, verify the service-to-ruleset association first.
- Defaults are templates. They do not process events until you create an actual service or action instance.

### Next steps

- Learn the core concepts: Rules.
- If you need to enrich events, see Set Property.

## Store and Forward

Once WinSyslog receives an event through an input service and it matches a rule, actions can store it locally and forward it to downstream systems.

### Where to configure

- Actions is the entry point for all output and notification options.

### Common storage targets

- Flat files: Write to File
- Databases: Write to Database
- Windows Event Log: Write to Event Log

### Database setup paths

- Default supported schema: Quick Start: Write Syslog Messages to Microsoft SQL Server
- Custom schema integration: Tutorial: Integrate WinSyslog with a Custom Database Schema

### Common forwarding targets

- Syslog: Forward Syslog
- SETP: Forward via SETP
- Email: Forward via Email
- Interactive viewing: forward messages to InterActive SyslogViewer if you want to see incoming data live in the Interactive Syslog Viewer.

### Related analysis component

- For reviewing stored logs later, Adiscon LogAnalyzer works with file- and database-based storage.
- Users who prefer a simpler standalone tool for direct log file review may also consider the third-party log viewer Retrospective by centeractive.

### Recommended setup path

1. Start with one storage action (file or database) to verify your rules.
2. Add a forwarding action after you are confident filtering is correct.
3. If you forward to a remote target, test connectivity and credentials early.

### Next steps

- If you need reliable intake on the input side, see RELP Listener service.

## Operate and Troubleshoot

After initial setup, most operational work is validating inputs, tuning rules, and diagnosing why something did or did not happen.

### Quick checklist

1. Confirm the WinSyslog service is running (Windows Services / *services.msc*).
2. Confirm your input service is enabled and attached to the intended ruleset.
3. Confirm your rule order and filters match what you expect.
4. Add a temporary Write to File action to see raw output quickly.

### Testing tools

- To validate syslog reception, use the WinSyslog Configuration Client menu *Tools* and run *Send Syslog Test Message* (see **`Send Syslog Test Message`**).

### Where to look next

- Input issues: Services
- Matching issues: Filter conditions
- Output issues: Actions
- UI basics and defaults vs. instances: Configuring WinSyslog

# Tutorials

This section contains task-oriented tutorials for common WinSyslog configuration tasks.

## Tutorial: Create a Simple Syslog Server and Write to a File

Use this tutorial when you want WinSyslog to receive syslog messages and store them in a local text file.

In this manual, **input** is the clearest plain-language concept, while the configured object is a **service**. In the GUI, that specific input service is named `Syslog server`.

### Goal

At the end of this procedure, WinSyslog will:

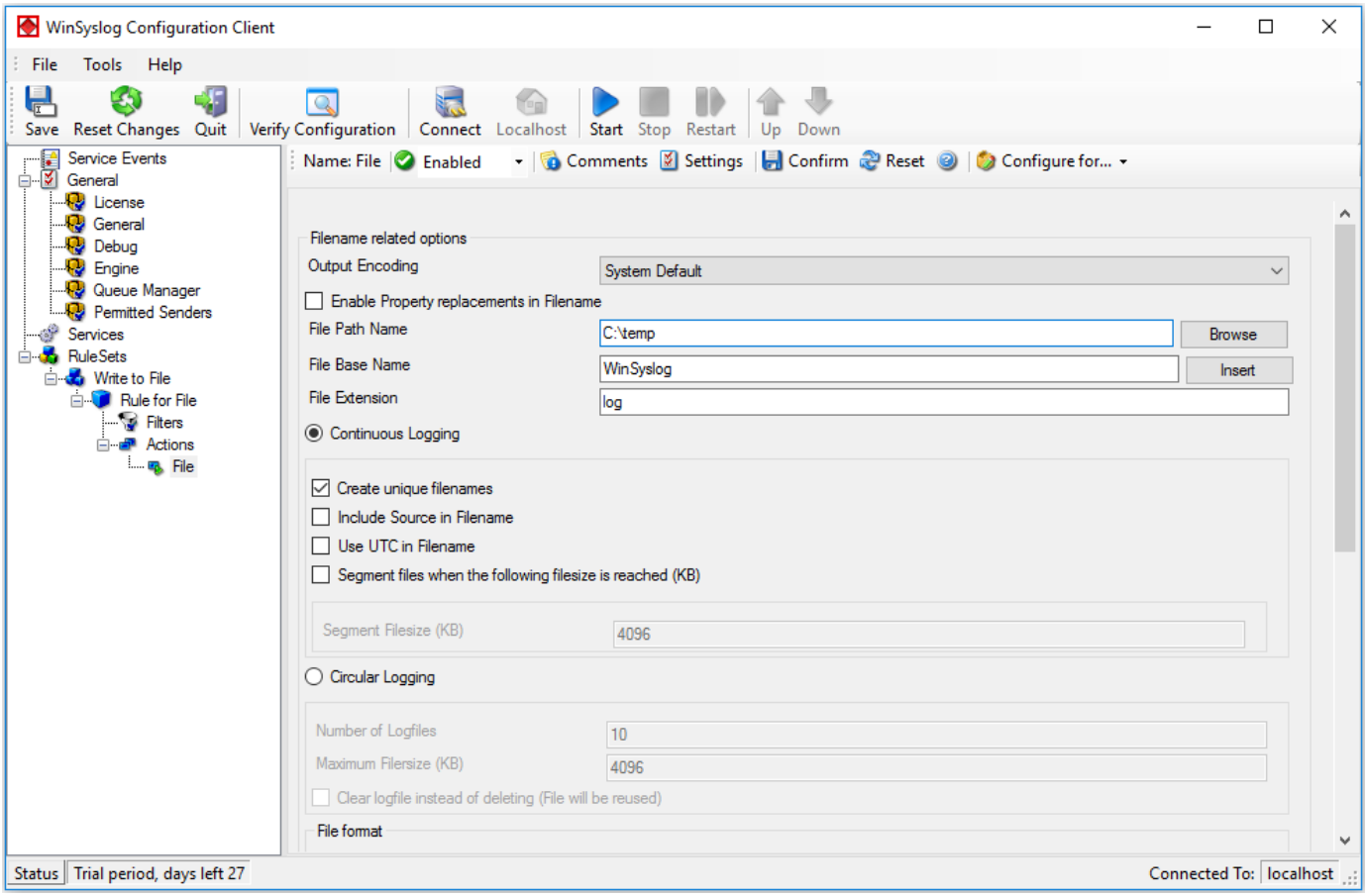
- listen for incoming syslog messages
- pass them through a ruleset
- write matching messages to a file on disk

### Prerequisites

- A writable target directory for log files
- At least one ruleset for incoming messages
- An input service that will be attached to that ruleset

### Steps

1. Create or choose a ruleset.
    - In the WinSyslog Configuration Client, create a ruleset for the messages you want to store.
    - If this is your first setup, you can also reuse the ruleset from Creating an Initial Configuration.
2. Add a **Write to File** action.
    - Inside that ruleset, add a Write to File action.
    - Open the file logging settings.




3. Configure the target file.
    - Set **File Path Name** to the directory where WinSyslog should write the files.
    - Set **File Base Name** to the logical file name prefix.
    - Keep the default extension unless you need something specific.
4. Decide how files should be created.
    - Use daily unique filenames if you want date-based rotation.
    - Use a continuous filename if another process expects one stable file name.
    - Enable **Include Source in Filename** only if you explicitly want separate files per sender.
5. Attach message intake to the ruleset.
    - Ensure at least one input service, for example a Syslog server service, is bound to the ruleset that contains the file action.
6. Save the configuration and restart the WinSyslog service if required.

## Verification

1. Send a test message with **Tools -> Send Syslog Test Message**.
2. Open the configured directory.
3. Confirm that WinSyslog created the expected log file.
4. Open the file with a tool that does not lock it aggressively, such as Notepad.

## Next step

If file logging works, continue with:

- Creating an Initial Configuration
- Write to File action reference
- Store and forward

# Quick Start: Write Syslog Messages to Microsoft SQL Server

Use this tutorial when you want a hands-on setup that turns one known syslog message into rows in Microsoft SQL Server with a reproducible table and field mapping.

## Question

How do I quickly write WinSyslog syslog messages to Microsoft SQL Server?

## Answer

Create one ODBC **System DSN**, point WinSyslog at one SQL Server table, map the syslog properties to the target columns, send one test message, and verify the row in SQL Server. This tutorial uses a fixed demo database so you can reproduce the result immediately.

## At a glance

- Database: `WinSyslogDemo`
- Table: `dbo.WinSyslogIncoming`
- DSN: `WinSyslogDemo`
- Input: one RFC 5424 syslog message
- Output: one row with raw and structured message columns

## Goal

At the end of this procedure, WinSyslog will:

- receive a test syslog message
- write that message into Microsoft SQL Server through an ODBC **System DSN**
- store both the raw message and a structured view of the same event in a simple database table

## What you will build

- a SQL Server database named `WinSyslogDemo`
- a table named `dbo.WinSyslogIncoming`
- an ODBC **System DSN** named `WinSyslogDemo`
- a WinSyslog ruleset with one **Write to Database** action
- a repeatable test row you can query in SQL Server

## Sample message and output

This quick start uses a single RFC 5424-style syslog message so the database row is easy to recognize.

Sample input message:

```
<134>1 2026-04-01T08:15:00Z fw01 sshd 1234 ID47 - Accepted password for alice from 192.0.2.10 port 55221
```

The example table stores the same event twice:

- once as the raw message
- once as the parsed message text and header fields

The mapping is intentionally small and explicit:

- `timereported` becomes `received_at`
- `source` becomes `source_host`
- `syslogfacility_text` becomes `facility_text`
- `syslogpriority_text` becomes `severity_text`
- `syslogappname` becomes `app_name`
- `rawsyslogmsg` becomes `raw_message`
- `msgPropertyDescribed` becomes `message_text`

Expected output in SQL Server:

```
received_at   = 2026-04-01 08:15:00
source_host   = fw01
facility_text = Local0
severity_text = Informational
app_name      = sshd
raw_message   = <134>1 2026-04-01T08:15:00Z fw01 sshd 1234 ID47 - Accepted password for alice from 192.0.2.10 port 55221
message_text  = Accepted password for alice from 192.0.2.10 port 55221
```

## Prerequisites

- Microsoft SQL Server
- SQL Server Management Studio (SSMS) if you want the GUI-based SQL workflow
- Microsoft ODBC Driver 18 for SQL Server, or a compatible Microsoft SQL Server ODBC driver installed on the WinSyslog host
- Database credentials with permission to connect, insert rows, and create tables
- Access to the WinSyslog configuration client on the host where WinSyslog runs

## Required downloads

If you do not already have the Microsoft components installed, use these official download pages before you start the tutorial:

- Microsoft SQL Server downloads

  Use the Express edition on this page if you need a free local SQL Server instance for the demo database.
- SQL Server Management Studio

  Use SSMS if you want to create the database and table in a GUI. The article includes the current direct download link.
- Microsoft ODBC Driver for SQL Server

  WinSyslog writes to SQL Server through ODBC, so install the driver that matches the WinSyslog host architecture.

## Steps

1. Create the target database and table in SQL Server.

   - Open SSMS and connect to the SQL Server instance that should receive the WinSyslog data.
   - Run the following SQL script to create the demo database and table:

```sql
IF DB_ID(N'WinSyslogDemo') IS NULL
BEGIN
    CREATE DATABASE WinSyslogDemo;
END
GO

USE WinSyslogDemo;
GO

IF OBJECT_ID(N'dbo.WinSyslogIncoming', N'U') IS NULL
BEGIN
    CREATE TABLE dbo.WinSyslogIncoming (
        received_at datetime2(0) NOT NULL,
        source_host nvarchar(255) NOT NULL,
        facility_text nvarchar(32) NULL,
        severity_text nvarchar(32) NULL,
        app_name nvarchar(128) NULL,
        raw_message nvarchar(max) NULL,
        message_text nvarchar(max) NOT NULL
    );
END
GO
```

If you prefer the command line, save the same script locally as `create-winsyslog-demo-table.sql` and run it with `sqlcmd`:

```
sqlcmd -S localhost -E -i .\create-winsyslog-demo-table.sql
```

If you already use the PowerShell `SqlServer` module, this is the same step with `Invoke-Sqlcmd`:

```
Invoke-Sqlcmd -ServerInstance localhost -InputFile .\create-winsyslog-demo-table.sql
```

2. Create and test an ODBC **System DSN** on the WinSyslog host.

   If you want to create the DSN from PowerShell instead of the GUI, run the helper script below in an elevated PowerShell session on the WinSyslog host. It creates a System DSN named `WinSyslogDemo` and can be adjusted for a different server or driver if needed.

```powershell
<#
.SYNOPSIS
Creates the WinSyslogDemo ODBC System DSN for the WinSyslog quick start.

.DESCRIPTION
Run this script in an elevated PowerShell session on the WinSyslog host.
It creates or replaces a System DSN by using the built-in WDAC ODBC cmdlets.

The script keeps credentials out of the DSN. Use -UseTrustedConnection only if
you want the DSN itself to use Windows authentication for connection tests.
#>

[CmdletBinding()]
param(
    [string]$Name = 'WinSyslogDemo',
    [string]$Server = 'localhost',
    [string]$Database = 'WinSyslogDemo',
    [string]$DriverName = 'ODBC Driver 18 for SQL Server',
    [ValidateSet('32-bit', '64-bit')]
    [string]$Platform = '64-bit',
    [switch]$UseTrustedConnection
)

$dsnProperties = @(
    "Server=$Server",
    "Database=$Database"
)

if ($UseTrustedConnection) {
    $dsnProperties += 'Trusted_Connection=Yes'
    $dsnProperties += 'TrustServerCertificate=Yes'
}

$existingDsn = Get-OdbcDsn -Name $Name -DsnType System -Platform $Platform -ErrorAction SilentlyContinue
if ($null -ne $existingDsn) {
    $existingDsn | Remove-OdbcDsn -ErrorAction Stop
}

Add-OdbcDsn -Name $Name `
    -DriverName $DriverName `
    -DsnType System `
    -Platform $Platform `
    -SetPropertyValue $dsnProperties `
    -PassThru `
    -ErrorAction Stop | Out-Null

Write-Host "Created System DSN '$Name' on $Platform with driver '$DriverName'."
Write-Host "Open Data Sources (ODBC), test the DSN, and then select it in WinSyslog."
```

After the script finishes, open **Data Sources (ODBC)** and test the new System DSN before you continue.

If the DSN uses Windows authentication, remember that WinSyslog normally runs under the default Windows `Local System` service account unless you changed it. A successful interactive admin test does not prove that the WinSyslog service can connect to SQL Server.

For a remote SQL Server, either grant SQL access to the WinSyslog service account context, change the WinSyslog service to run under an account that already has SQL access, or use SQL authentication instead.

The practical verification step is to restart the WinSyslog service, send a test message, and then check whether the row is inserted.

You can also run this small PowerShell connection test against the DSN:

```
$connection = [System.Data.Odbc.OdbcConnection]::new('DSN=WinSyslogDemo;')
$connection.Open()
$connection.Close()
Write-Host 'DSN connection test succeeded.'
```

If you prefer the GUI instead, use these steps:

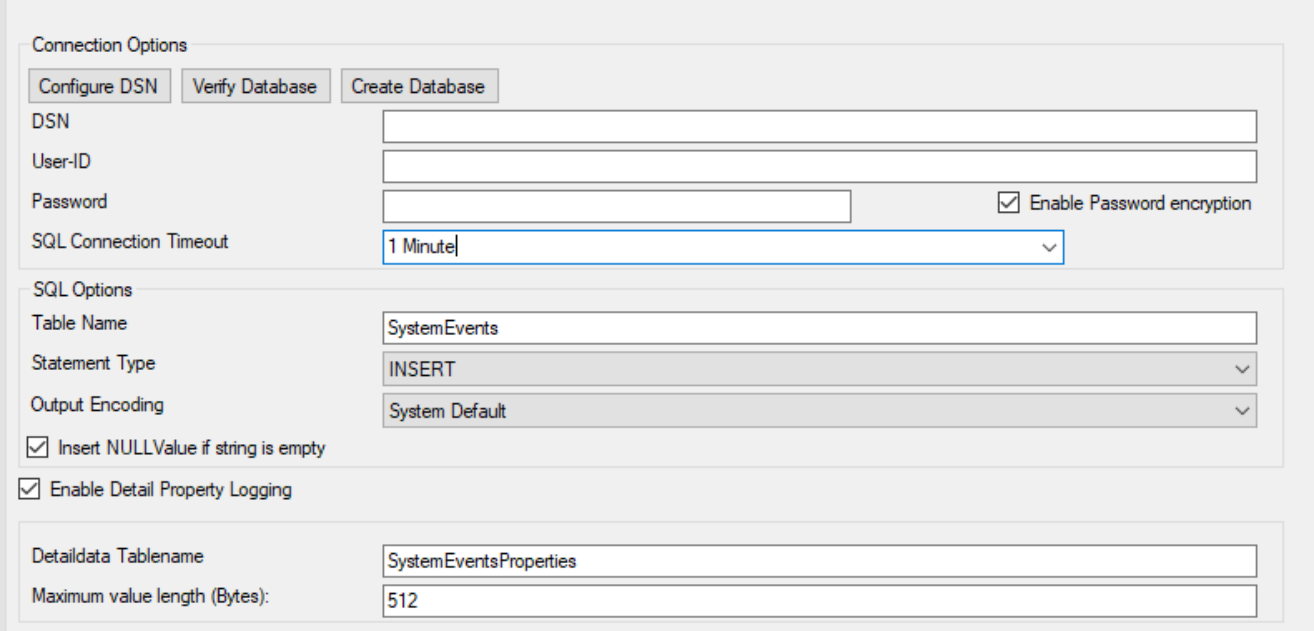
- Open **Data Sources (ODBC)** on the WinSyslog host.
- Create a new **System DSN** for SQL Server and name it `WinSyslogDemo`.
- Select the Microsoft SQL Server driver that matches your environment, for example Microsoft ODBC Driver 18 for SQL Server.
- Point the DSN to the SQL Server instance and database created in the previous step.
- Complete the DSN wizard and use its built-in connection test.

3. Create or choose the WinSyslog ruleset whose messages should be stored.

- If you are starting from scratch, create a new ruleset for the database action.
- Open the `Syslog server` service that should receive the test message and bind it to that same ruleset.
- Make sure that service is enabled before you send the first test message.
- If you already have a working ruleset, confirm that the receiving service is bound to the same ruleset that holds the database action.

4. Add a Write to Database action to that ruleset.
5. Configure the database action.

- Select the ODBC **System DSN** `WinSyslogDemo`.
- Enter the database credentials if the DSN or driver requires them.
- Set the table name to `dbo.WinSyslogIncoming`.
- Keep the SQL statement type at the normal `INSERT` path.
- Leave output encoding at `System Default` unless you know you need a different one.
- Do **not** use **Create Database** for this quick start because the table is created by the SQL script above.

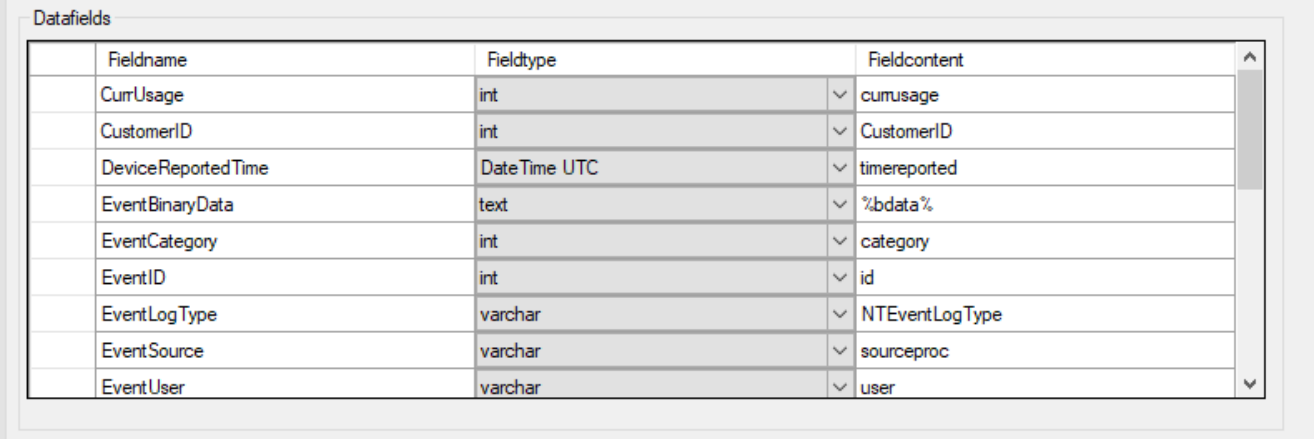



*The connection tab is the main place to select the DSN, enter credentials, and verify the connection before you save the action.*

6. Configure the field list so it matches the demo table.

- Remove any fields that do not belong in the demo table.

- Use these mappings:
  - `received_at` -> `DateTime` -> `timereported`
  - `source_host` -> `varchar` -> `source`
  - `facility_text` -> `varchar` -> `syslogfacility_text`
  - `severity_text` -> `varchar` -> `syslogpriority_text`
  - `app_name` -> `varchar` -> `syslogappname`
  - `raw_message` -> `text` -> `rawsyslogmsg`
  - `message_text` -> `text` -> `msgPropertyDescribed`




*The datafields tab is where you turn one incoming syslog event into the structured columns shown in the example table.*

7. Save and apply the configuration.
   - Restart the WinSyslog service if your environment or workflow requires it.
8. Send a test message.
   - For the fastest first verification, use **Tools -> Send Syslog Test Message**.
   - To reproduce the sample row above, send an RFC 5424 syslog message with the same values from any syslog client.
9. Verify the inserted rows in SQL Server.
   - Open SSMS and query the demo table.
   - Confirm that the row contains the expected values.

```
SELECT TOP (10) *
FROM dbo.WinSyslogIncoming;
```

If you prefer the command line, run the same check with `sqlcmd`:

```
sqlcmd -S localhost -E -Q "SELECT TOP (10) * FROM dbo.WinSyslogIncoming;"
```

## Verification

1. The ODBC **System DSN** test succeeds.
2. The WinSyslog action connects successfully to the DSN.
3. A test message produces a new row in `dbo.WinSyslogIncoming`.
4. The row contains both the raw syslog line and the structured message columns.

## Common issues

- The DSN was created as a user DSN instead of a **System DSN**. See FAQ: Why Does My WinSyslog Database Setup Fail?.

- The DSN points to the wrong SQL Server instance or database. See FAQ: Why Does My WinSyslog Database Setup Fail?.
- The SQL Server account can connect but does not have permission to create rows. See FAQ: Why Does My WinSyslog Database Setup Fail?.
- The table name in WinSyslog does not match the SQL table name exactly. See FAQ: Why Does My WinSyslog Database Setup Fail?.
- The field list still contains the default schema rows instead of the demo table columns. See FAQ: Why Does My WinSyslog Database Setup Fail?.
- The message sender is not reaching the same ruleset that holds the database action. See FAQ: Why Does My WinSyslog Database Setup Fail?.
- The data type in the table is too short for the message text and truncation occurs. See Tutorial: Integrate WinSyslog with a Custom Database Schema.

### Next step

If this quick start works and you want to adapt the schema to your own application, continue with:

- Tutorial: Integrate WinSyslog with a Custom Database Schema

### See also

- ODBC Database Options
- Why Does My WinSyslog Database Setup Fail?
- Store and Forward
- Which Database Format Should I Use with WinSyslog?

## Tutorial: Integrate WinSyslog with a Custom Database Schema

Use this tutorial when WinSyslog should write into an existing database schema instead of the built-in `SystemEvents` layout.

If you want the fastest first success path, start with Quick Start: Write Syslog Messages to Microsoft SQL Server first and return here when you need to adapt the database layout to an existing application.

### Question

How do I write WinSyslog data into an existing custom SQL table?

### Answer

Use an ODBC **System DSN**, point WinSyslog at the existing table, replace the default field list with the columns from your schema, and verify the insert with a test message. This path is for existing databases, not for first-time setup.

### At a glance

- Use this page only when the schema already exists
- Keep the target table fixed and map each column deliberately
- Test the DSN before debugging WinSyslog
- Verify with one matching message and one SQL query

### Goal

At the end of this procedure, WinSyslog will write matching messages into your own destination table through an ODBC **System DSN**.

### When to choose this tutorial

Use this path when:

- your organization already has a fixed database schema

- another application expects specific column names or data types
- you want WinSyslog to feed an existing integration or reporting database

Do not use this path just because database logging exists. If you want the fastest supported setup or a ready-made example, use Quick Start: Write Syslog Messages to Microsoft SQL Server instead.

## What this tutorial does not do

This tutorial does not design your schema for you. WinSyslog can write to your table, but you must decide the table design, column definitions, indexes, retention strategy, and downstream reporting logic.

## Prerequisites

- A reachable database server and a tested ODBC **System DSN**
- A destination table that already exists
- Database credentials with permission to insert rows into that table
- A clear mapping from WinSyslog properties to the destination columns

## Example target table

The exact schema is up to you. For a Microsoft SQL Server example, a custom table could look like this:

```
CREATE TABLE dbo.IncomingSyslog (
    received_at datetime2 NOT NULL,
    source_host nvarchar(255) NOT NULL,
    facility_text nvarchar(32) NULL,
    severity_text nvarchar(32) NULL,
    app_name nvarchar(128) NULL,
    raw_message nvarchar(max) NULL,
    message_text nvarchar(max) NOT NULL
);
```

## Sample message and output

This example uses the same RFC 5424-style message as the quick start so you can compare the default and custom-schema paths directly.

Sample input message:

```
<134>1 2026-04-01T08:15:00Z fw01 sshd 1234 ID47 - Accepted password for alice from 192.0.2.10 port 55221
```

Expected output in the example custom table:

```
received_at   = 2026-04-01 08:15:00
source_host   = fw01
facility_text = Local0
severity_text = Informational
app_name      = sshd
raw_message   = <134>1 2026-04-01T08:15:00Z fw01 sshd 1234 ID47 - Accepted password for alice from 192.0.2.10 port 55221
message_text  = Accepted password for alice from 192.0.2.10 port 55221
```

## Steps

1. Review the destination schema before opening WinSyslog.
    - Confirm the exact table name.
    - Confirm each destination column name and data type.
    - Decide which WinSyslog property belongs in each column.
    - Decide whether you want to keep the raw message, the parsed message text, or both.
2. Create and test an ODBC **System DSN** for the target database.
    - The DSN must point to the database that already contains your destination table.

- Complete the DSN wizard and use its built-in test before you continue.
- If the DSN uses Windows authentication, remember that WinSyslog normally runs under the default Windows `Local System` service account unless you changed it. A DSN test that succeeds for the interactive admin user does not prove that the WinSyslog service can connect to the same SQL Server instance and database.
- For a remote SQL Server, either grant SQL access to the WinSyslog service account context, change the WinSyslog service to run under an account that already has SQL access, or use SQL authentication instead.

3. Create or choose the ruleset whose messages should be stored.

   - If you are starting from scratch, create a dedicated ruleset for the database action.
   - Bind the receiving service to that same ruleset before you test inserts.
   - If you already use an existing ruleset, confirm that the service that receives the message is bound to the ruleset that holds the database action.

4. Add a Write to Database action to that ruleset.
5. Configure the connection settings.

   - Select the ODBC **System DSN**.
   - Enter credentials if the DSN or driver requires them.
   - Use **Verify Database** to confirm connectivity.

6. Point the action to the existing destination table.

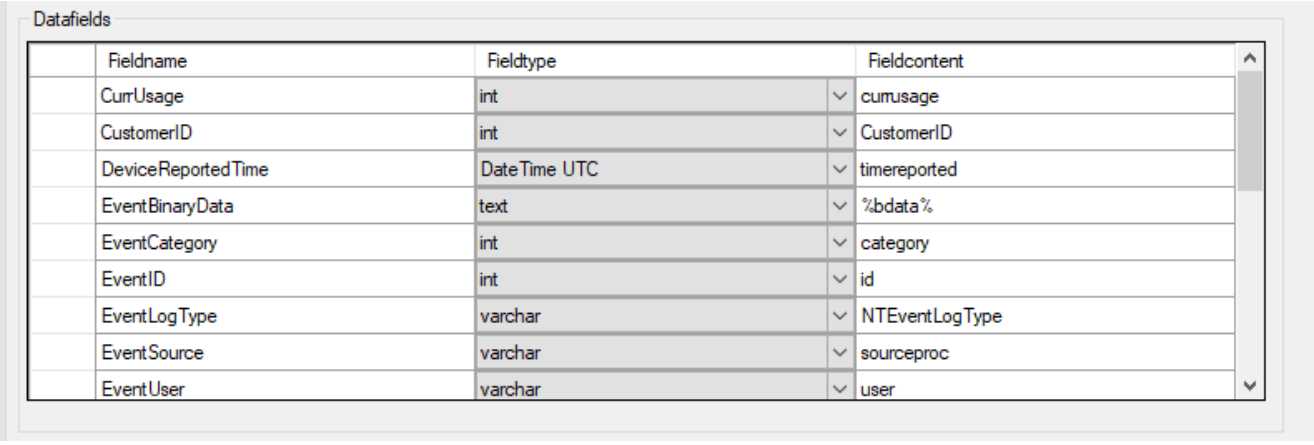
   - Enter the custom table name.
   - Do **not** use **Create Database** for this path unless you intentionally want the built-in default schema instead of your own table.

7. Replace the default field list with mappings that match your schema.

   For the example table above, a practical starting point is:

   - `received_at` -> `DateTime` -> `timereported`
   - `source_host` -> `varchar` -> `source`
   - `facility_text` -> `varchar` -> `syslogfacility_text`
   - `severity_text` -> `varchar` -> `syslogpriority_text`
   - `app_name` -> `varchar` -> `syslogappname`
   - `raw_message` -> `text` -> `rawsyslogmsg`
   - `message_text` -> `text` -> `msgPropertyDescribed`

   If a string column is shorter than the source property, use the property replacer to truncate or transform the value deliberately. For example, `%msg:1:200%` stores only the first 200 characters of the message.




   *The datafields tab is where you replace the default mapping with the column names, data types, and event properties that match your own schema.*

8. Save and apply the configuration.

   - Restart the WinSyslog service if your environment or workflow requires it.

9. Send or receive a matching test message.
10. Query the destination table and verify the inserted data.

.

```
SELECT TOP (10)
    received_at,
    source_host,
    facility_text,
    severity_text,
    app_name,
    raw_message,
    message_text
FROM dbo.IncomingSyslog;
```

### Verification

1. The ODBC **System DSN** test succeeds.
2. The action's **Verify Database** button succeeds.
3. A test message inserts a row into the existing custom table.
4. Each value appears in the expected destination column with the expected data type and length.

### Common issues

- Leaving the default field list unchanged while targeting a custom table
- Using the wrong field type for a destination column
- Forgetting that long text may not fit into a short `varchar` column
- Clicking **Create Database** even though the goal is an existing custom schema
- Assuming Adiscon tools that expect the default schema will continue to work unchanged against the custom table
- Forgetting to test the DSN outside WinSyslog before debugging the action
- Mapping `msg` to a short column when the real message text is longer than the destination can hold

### Next step

If you later decide that you need the built-in Adiscon table layout instead of your own schema, switch to Quick Start: Write Syslog Messages to Microsoft SQL Server.

### See also

- ODBC Database Options
- Why Does My WinSyslog Database Setup Fail?
- Which Database Format Should I Use with WinSyslog?

## Tutorial: Prepare WinSyslog Data for Adiscon LogAnalyzer

Use this tutorial when WinSyslog should write data that you want to review later in Adiscon LogAnalyzer.

### Goal

At the end of this procedure, WinSyslog will write messages into a database that LogAnalyzer can open.

### Recommended path

For WinSyslog, the recommended LogAnalyzer path is database-backed storage. This avoids file-parser dependencies and is the most stable integration path in the current manual.

LogAnalyzer is the browser-based review component for stored data. It is not the WinSyslog service administration interface. For that distinction, see How Do Remote Administration and Browser-Based Log Review Fit Together?.

### Prerequisites

- A reachable database server
- An ODBC **System DSN** on the WinSyslog host
- WinSyslog access to the target database
- A LogAnalyzer deployment that is ready to connect to the same database

### Steps

1. Complete Quick Start: Write Syslog Messages to Microsoft SQL Server so WinSyslog writes matching messages into the database.
2. Open **`../shared/tutorials/loganalyzer-setup-and-use`**.
3. Configure LogAnalyzer to use the same database as its data source.
4. Send or receive one or more matching messages.

### Verification

1. Confirm that WinSyslog writes rows into the target table.
2. Open the configured source in LogAnalyzer.
3. Verify that the stored messages appear there.

### Next step

If you need to refine which messages are stored before they appear in LogAnalyzer, continue with:

- Process and Filter
- Quick Start: Write Syslog Messages to Microsoft SQL Server

### Complementary Tool: Retrospective

If you need a simpler way to search and analyze log files directly instead of using a web-based database setup, another available option is the third-party log viewer Retrospective by centeractive. It is suited for direct log file review without requiring a central database setup.

## Tutorial: Export the Configuration and Create a Debug Log

Use this tutorial when you need WinSyslog configuration data and a debug log for troubleshooting or for a support case.

### Goal

At the end of this procedure, you will have:

- a text-based export of the WinSyslog configuration
- a debug log file that captures service behavior

### Prerequisites

- Access to the WinSyslog Configuration Client
- Permission to restart the WinSyslog service if needed
- A writable location for the exported settings and debug log

### Steps

1. Export the WinSyslog configuration.

    - Open the WinSyslog Configuration Client.
    - Go to **Computer -> Export Settings to Registry File**.
    - Export the settings as a text-based registry file.

2. Save the export to a known location.
    - Use a descriptive file name such as `winsyslog-config.reg`.
    - Do not use the binary export format.
3. Enable debug logging.
    - Open Debug under **General**.
    - Enable debug output into file.
    - Set a full path and file name for the debug log.
    - Start with **Errors & Warnings** and **Minimum Debug Output** unless support asks for a higher level.
4. Restart the WinSyslog service if required.
5. Reproduce the problem or run the test scenario.
6. Disable debug logging again after capture.
    - Debug logging increases system load and should not stay enabled during normal operation.
7. Package the files for review or support.
    - Compress the exported registry file and the debug log file into one ZIP archive.

### Verification

1. Confirm that the registry export file exists and is readable as text.
2. Confirm that the debug log file exists and contains recent entries from the test period.
3. If the debug log stays empty, verify that the service was restarted after enabling debug output.

### Next step

If you need to send the data to Adiscon, continue with:

- How to Export WinSyslog Settings for a Support Call
- How do I contact Adiscon sales?

## Tutorial: Configure a SETP Server Service

Use this tutorial when another Adiscon product should forward events to WinSyslog via SETP.

In this manual, SETP reception is described as a **service**. In the GUI, that service is named `SETP Server`.

### Goal

At the end of this procedure, WinSyslog will listen for incoming SETP traffic and pass received events into a ruleset.

### Prerequisites

- A sending Adiscon product that supports SETP
- A ruleset with at least one visible action, such as file logging
- Port and TLS settings agreed between sender and receiver

### Steps

1. Create or choose a ruleset for incoming SETP events.
2. Add at least one action to that ruleset so received events are visible.
    - For a first test, a Write to File action is the simplest choice.
3. Add a SETP Server service.
4. Configure the input settings.
    - Keep the default port unless your environment requires a different one.
    - Bind to a specific IP address only if you need that behavior.

    - Enable TLS only if the sending side is configured for TLS as well.
5. Bind the service to the ruleset.
6. Save the configuration and restart the WinSyslog service if required.
7. Configure the sending system to forward to this WinSyslog instance over SETP.

### Verification

1. Send one or more events from the configured sender.
2. Confirm that the events reach the ruleset on the WinSyslog side.
3. If you used a temporary file action, verify that the events appear in the file.

### Next step

If the receiver works, continue with:

- SETP Server
- Receive Logs
- What Is the Difference Between SETP and Syslog?

## Tutorial: Forward Messages to Another Syslog Server

Use this tutorial when WinSyslog should relay selected messages to another syslog receiver.

### Goal

At the end of this procedure, WinSyslog will forward matching messages from a ruleset to a remote syslog server.

### Prerequisites

- The destination host name or IP address
- The target port and transport mode expected by the receiver
- TLS details if the target requires encrypted transport

### Steps

1. Create or choose the ruleset whose messages should be forwarded.
2. Add a Forward Syslog action to that ruleset.
3. Configure the remote target.
    - Enter the destination host name or IP address.
    - Set the destination port.
    - Choose the transport mode that matches the receiver.
4. Keep protocol choices conservative.
    - Use UDP only if message loss is acceptable.
    - Prefer TCP-based modes when reliable delivery matters.
    - Enable TLS only when the target is configured for it.
5. Save the configuration and restart the WinSyslog service if required.

### Verification

1. Send a test message into WinSyslog.
2. Confirm that the target syslog server receives the forwarded event.
3. If forwarding fails, verify host, port, protocol mode, and TLS settings on both sides.

### Next step

If forwarding works, continue with:

- Syslog Forwarding
- Store and Forward
- How Do I Handle Non-Standard Syslog Messages from Unix or Linux Systems?

## Tutorial: Send Matching Messages by Email

Use this tutorial when WinSyslog should send selected messages to an email recipient.

### Goal

At the end of this procedure, WinSyslog will send matching events as email messages through a configured SMTP server.

### Prerequisites

- A reachable SMTP server
- A sender address accepted by that server
- At least one target recipient address

### Steps

1. Create or choose the ruleset that should trigger email alerts.
2. Add a Send Email action.
3. Configure the SMTP connection.
    - Enter the mail server name or IP address.
    - Set the SMTP port.
    - Enable authentication if your server requires it.
    - Enable TLS or SSL only if the mail server supports it.
4. Configure sender and recipient fields.
5. Keep the first test simple.
    - Start with one recipient and a broad but temporary matching rule.
6. Save the configuration and restart the WinSyslog service if required.

### Verification

1. Trigger a matching event or send a controlled test message.
2. Confirm that the email reaches the recipient mailbox.
3. If it does not arrive, verify SMTP connectivity, authentication, and any mail relay restrictions.

### Next step

If the basic alert works, narrow the matching rule and review:

- Send Email
- Store and Forward

# InterActive SyslogViewer

InterActive SyslogViewer is a tool that lets you review your syslog data very easy. It is a separate syslog server, that simply displays all incoming data. By this you can see directly what is happening.

# InterActive SyslogViewer

## Features

**Fast and Easy syslog Viewing**

The SyslogViewer allows you to directly view and review syslog messages. Therefore you can react much better on occurring problems or check if everything is OK.

**Review stored logs from a database**

You can as well directly review log entries in a database. Simply enter the login details and that’s it. You can then review your logs and even filter the view. That helps you to find the important data in an easy way.

**Export selected data**

You can export selected data for further manual processing, like sending an email to your colleague for informing them about what is happening.

## Requirements

Windows 10, Windows 11, and Windows Server 2016/2019/2022/2025 (and newer versions).

SyslogViewer requires Microsoft .NET Framework. The installer will guide you through required prerequisites.

Hardware requirements: - 256 MB RAM or more (depending on workload and database connectivity)

# Options & Configuration

InterActive SyslogViewer is an add-on to the WinSyslog and WinSyslog. **Please note that it is a utility program, with a primary focus on real-time** troubleshooting.**

InterActive SyslogViewer is **not** meant to continuously monitor a system. This is what the service is designed for. While InterActive SyslogViewer allows to view current syslog traffic, the service should be used for all other purposes, like creating log files.

## Launching InterActive SyslogViewer

To run the InterActive SyslogViewer, click the “SyslogViewer” icon present in the Programs Folder -> WinSyslog/WinSyslog located in the Start menu.

It can also be launched from the command prompt:

- Open a Command Prompt window
- Change to the drive and directory where the WinSyslog is installed
- Type “InterActive SyslogViewer.exe” and hit enter

Available Command Line parameters are:

```
/? => Show Options
/autolisten => Start Syslog server automatically
/port=10514 => Overwrites the configured port
/windowpos 0,0,512,800 => Sets default window positions
```

## Using InterActive SyslogViewer

InterActive SyslogViewer is an add-on to WinSyslog and WinSyslog.

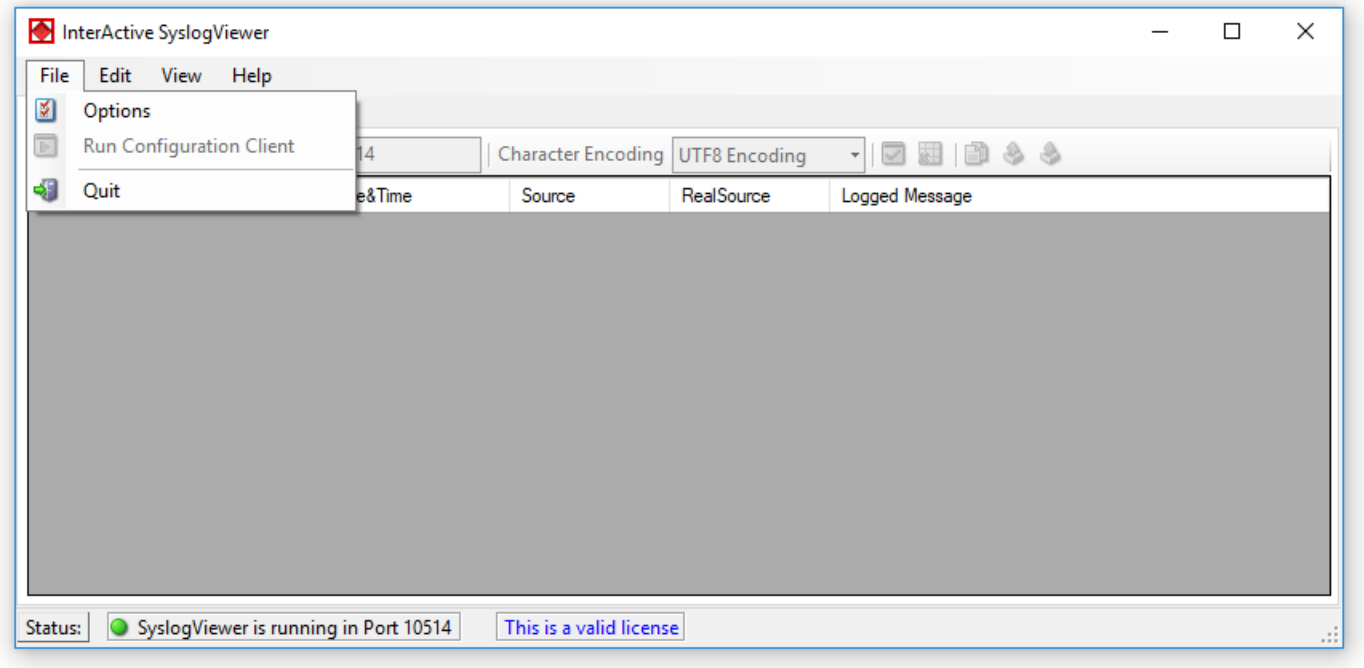
**Please note that it is a utility program with a primary focus on real-time troubleshooting**.

Interactive Syslog server is not meant to continuously monitor a system. This is what the service is designed for. While Interactive Server allows to view current Syslog traffic, the service should be used for all other purposes, like creating log files.

## Options & Menus

Please find more information about the different menus and options in the respective sub-category.

### File Menu

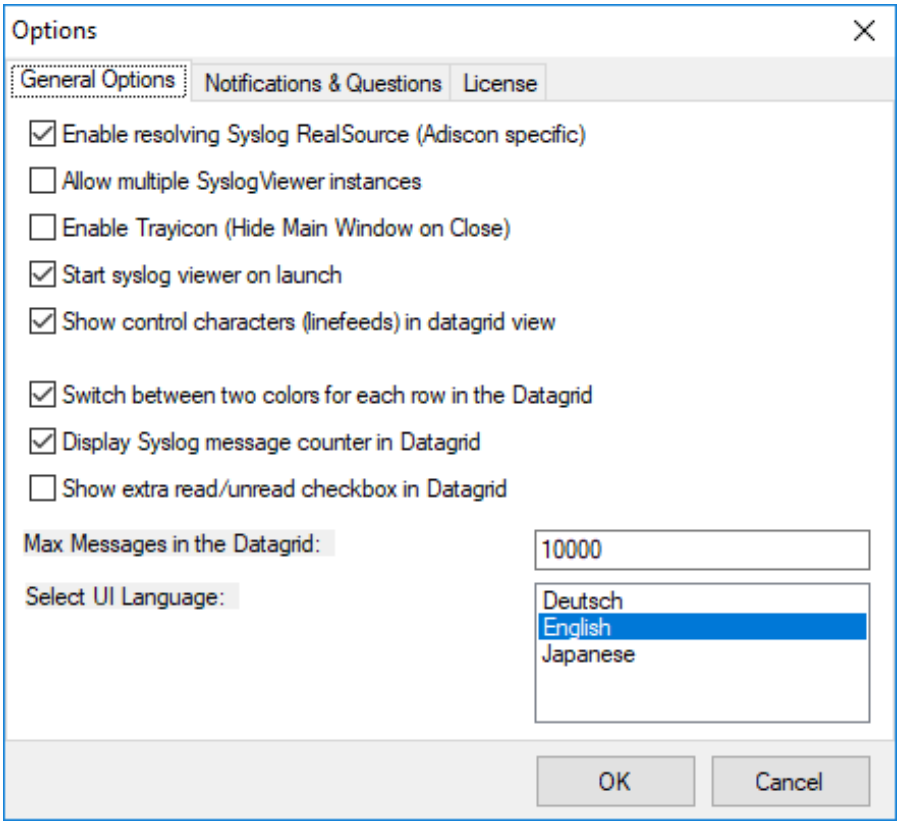



- InterActive SyslogViewer - File Menu*

### Options

This will open the Options dialog. Please see the sub-chapters for more details on this.

### General Options




- InterActive SyslogViewer - General Options Tab*

**Enable Resolving Syslog RealSource (Adiscon specific)**

With this option enabled, you can see the real source in multiply forwarded messages. That means, you can see the system that forwarded the message and the system where the message originates from.

**Allow multiple SyslogViewer instances**

You can have multiple instances of the InterActive SyslogViewer by activating this option. This allows you to have multiple forwarding servers sending on different ports and receive their messages separately.

**Enable Trayicon (Hide Main Windows on Close)**

Enable this to have a tray icon. This enables a soft-close. InterActive SyslogViewer will stay active, but the window will be completely hidden except the tray icon. By double-clicking on the icon, the window will show again.

**Start SyslogViewer on launch**

Enable this to start the Syslog server directly when starting InterActive SyslogViewer.

**Show control characters (line feeds) in data grid view**

When enabled, you will see control characters like line feeds in the data grid as well.

**Switch between two colors for each row in the data grid**

To have a better overview over the syslog data, activate this option.

**Display Syslog message counter in the data grid**

You can enable a counter by checking the box here. It will count further, even if the maximum of messages is already exceeded.

**Show extra read/unread checkbox in the data grid**

If enabled, an additional checkbox is added for each record in the data grid that can be marked as checked.

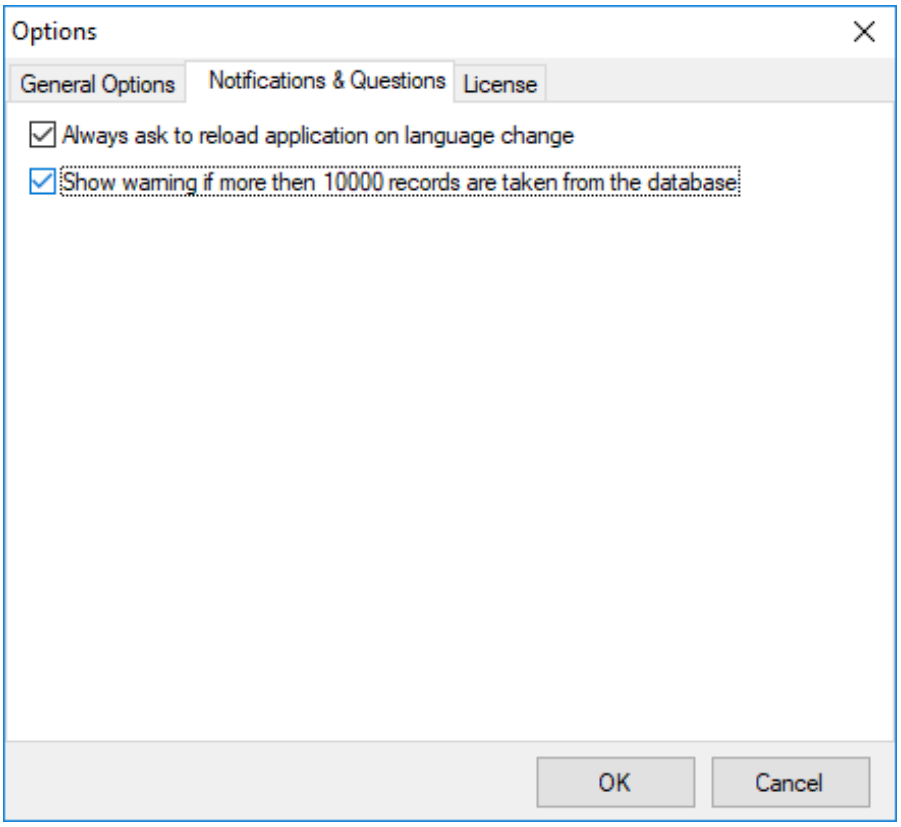
**Max Messages in the data grid**

Here you can adjust the maximum messages that will be available in the data grid. By increasing this value, you can store more messages for direct review. Please note, that increasing the maximum number of messages will have a severe impact on your memory.

**Select UI Language**

Here you can choose your favorite language for the InterActive SyslogViewer. By default it is English. You can choose German or Japanese as well.

### Notifications & Questions Tab




- InterActive SyslogViewer - Notifications & Questions Tab*

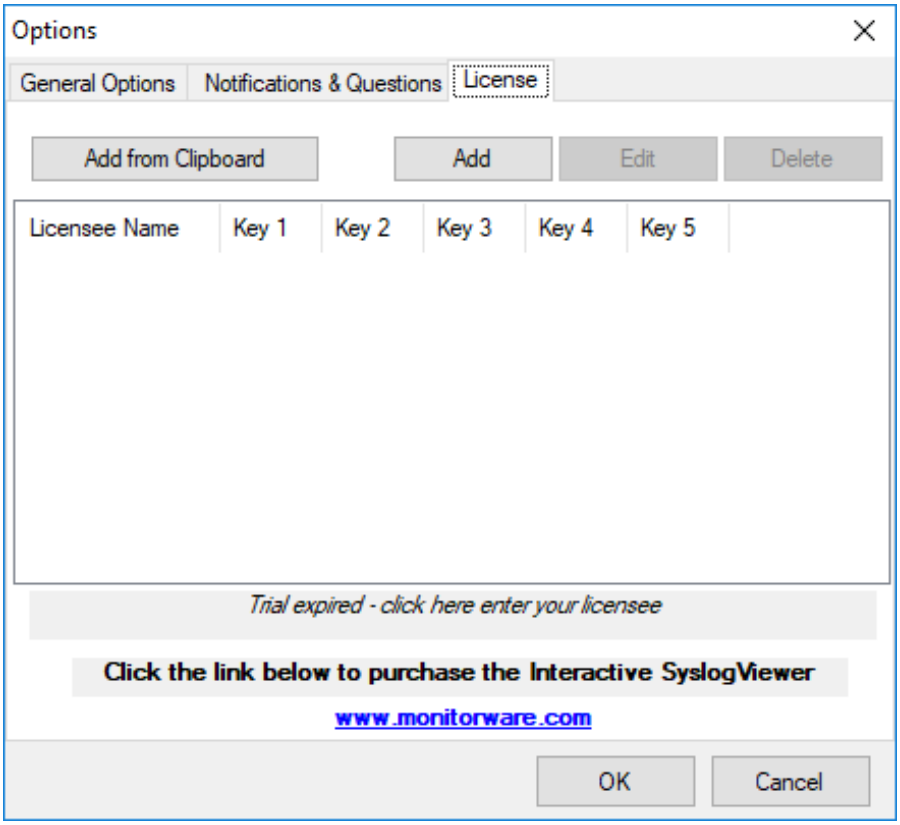
**Always ask to reload application on language change**

While the box is checked, InterActive SyslogViewer will ask to reload the application on a language change. This is, because the language file can only be loaded while starting the application and not while it is running.

**Show warning if more than 10000 records are taken from the database**

By activating this option, you will be warned, if the records in the database are just too much. This is to prevent the machine from receiving too much load. Polling lots of messages from a database can have a severe impact on the performance of the machine.

### License Tab




- InterActive SyslogViewer - License Tab*

Here you can insert the license. You have several options:

**Add from Clipboard**

This will insert the license you have currently on your clipboard.

**Add**

This button is to manually add a license manually. A new window will open, which shows you the form for entering the license information. This consists of a license name and five blocks of numbers.

**Edit**

Once a license is entered, it can be changed afterwards. This is done with this button. Mark the license you want to edit and click the button. A window will open which looks just like when adding a license, but the marked license details are inserted already. You can edit every field separately.

**Delete**

If a license is not needed anymore, you can delete it from the license screen. Mark the license and hit the button. The license will be deleted directly.

**Please note**, that the screen will give you additional information. You have

an overview of the licenses used and if not entered correctly it will show how long your trial period still is.

**run configuration client**

this option will open the configuration client of WinSyslog/WinSyslog. here you can do detail configuration of the service.

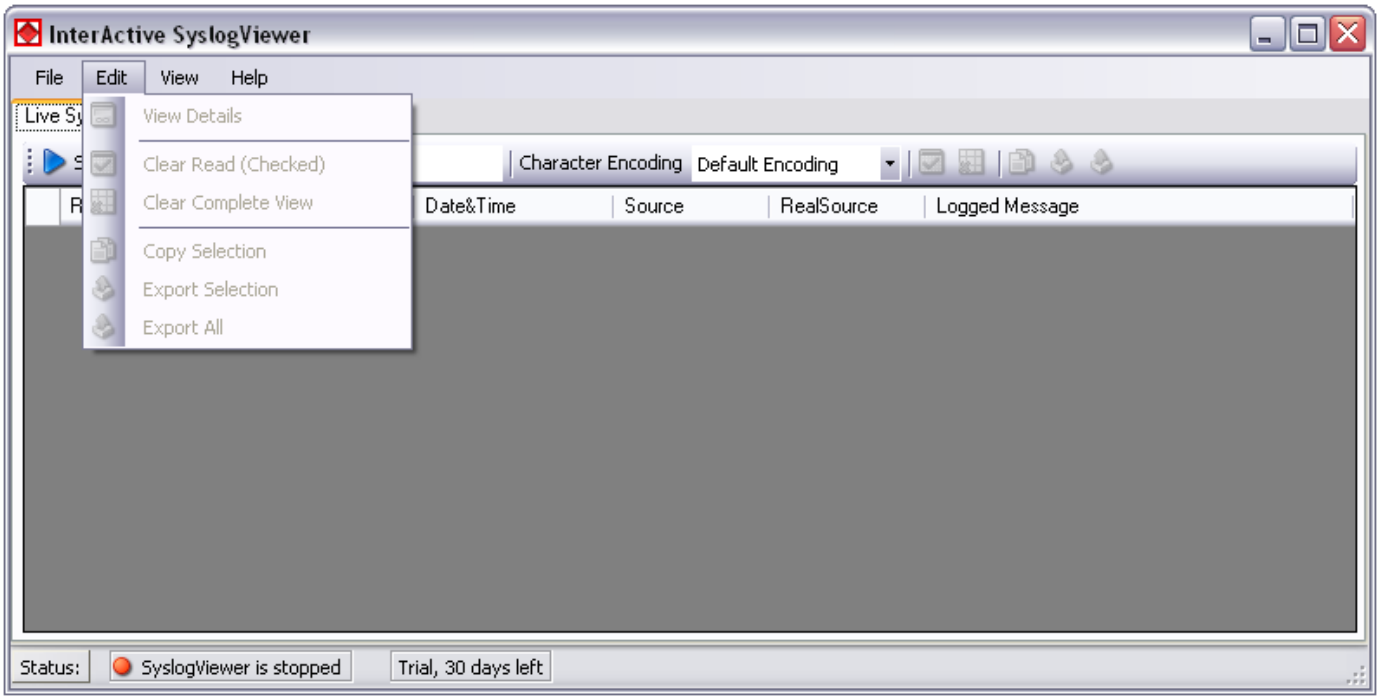
**minimize to tray**

this will minimize the interactive syslogviewer window and remove it from the taskbar. you can open it again by double-clicking on the icon in the system tray.

**quit**

by clicking here, interactive syslogviewer stops receiving data and it will close the application.

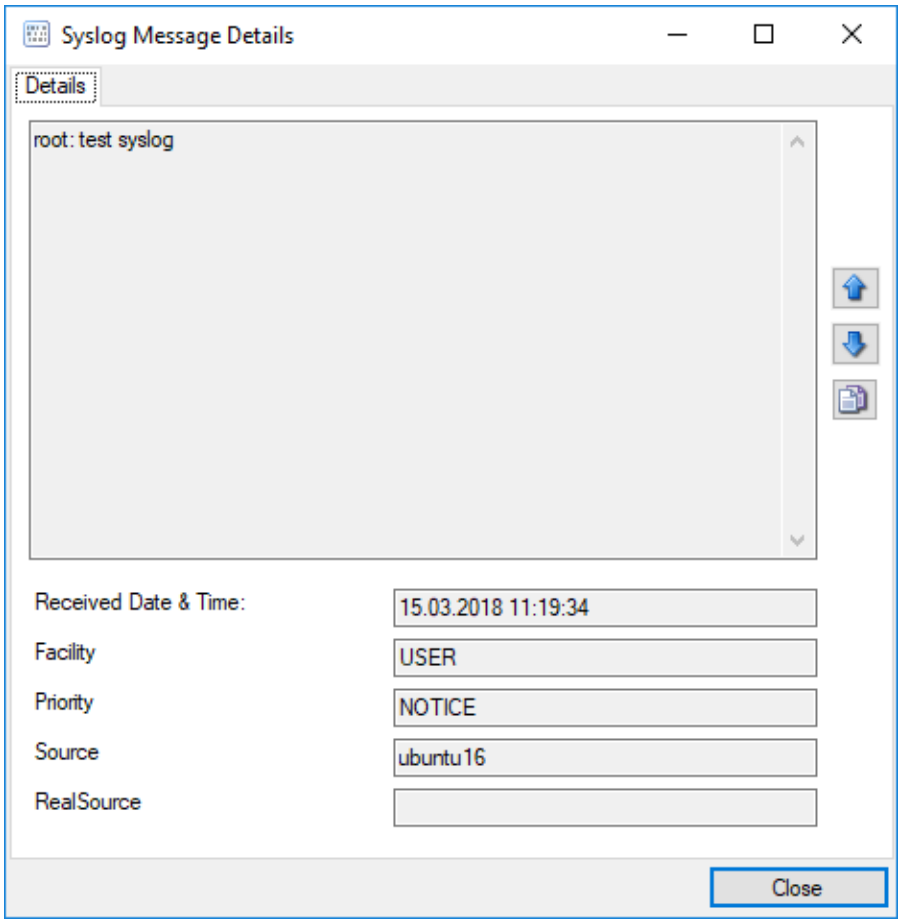
### Edit Menu




- InterActive SyslogViewer - Edit Menu*

**View Details**

When using this option, another window will open up, which shows the details of this event in a more readable view. This could look like this:




- InterActive SyslogViewer Syslog - Message Details*

**Clear Read (Checked)**

By activating this, you can clear the checkboxes of the items your marked as read.

**Clear Complete View**

This option will clear the screen and remove all received data from the view.

**Copy Selection**

Having selected one or multiple entries, you can copy them using this function.

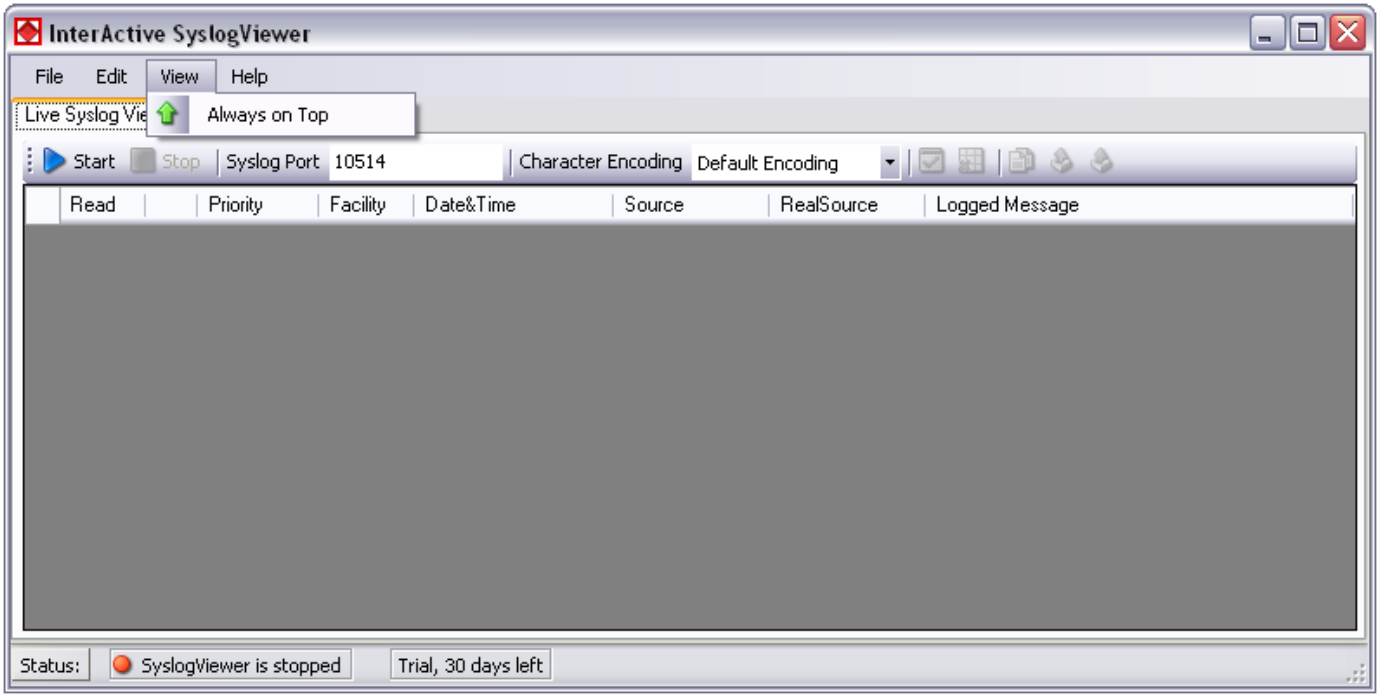
**Export Selection**

Instead of copying you can extract the selected data into a text file.

**Export All**

Or you directly export all the data that is currently in the list.

## View Menu

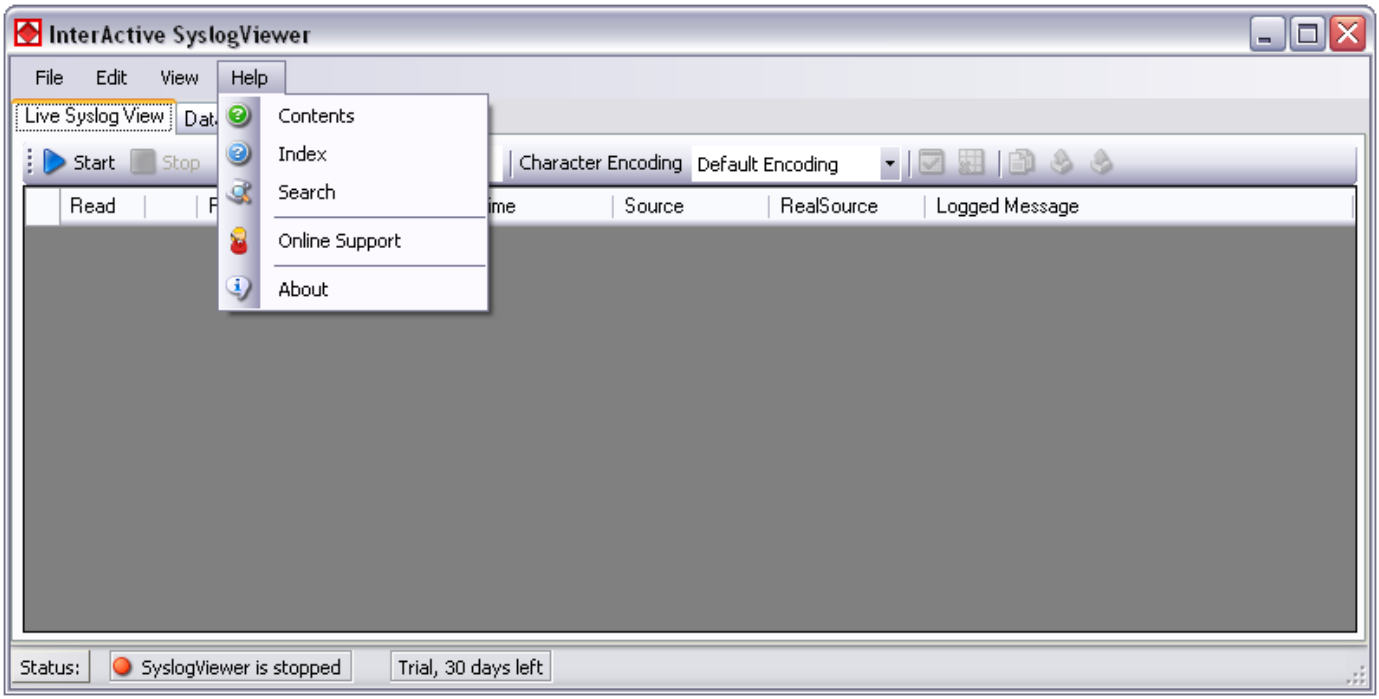



- InterActive SyslogViewer Syslog - View Menu*

**Always on Top**

This option is very self-explanatory. While activated, the InterActive SyslogViewer window will stay on top of all other applications, so you will have all incoming log data directly in your point of view.

### Help Menu




- InterActive SyslogViewer Syslog - Help Menu*

**Contents**

Show the manual.

**Index**

Show the manual index.

**Search**

Search the manual.

**Online Support**

By clicking here, a browser window will open and you will be directed to our support website.

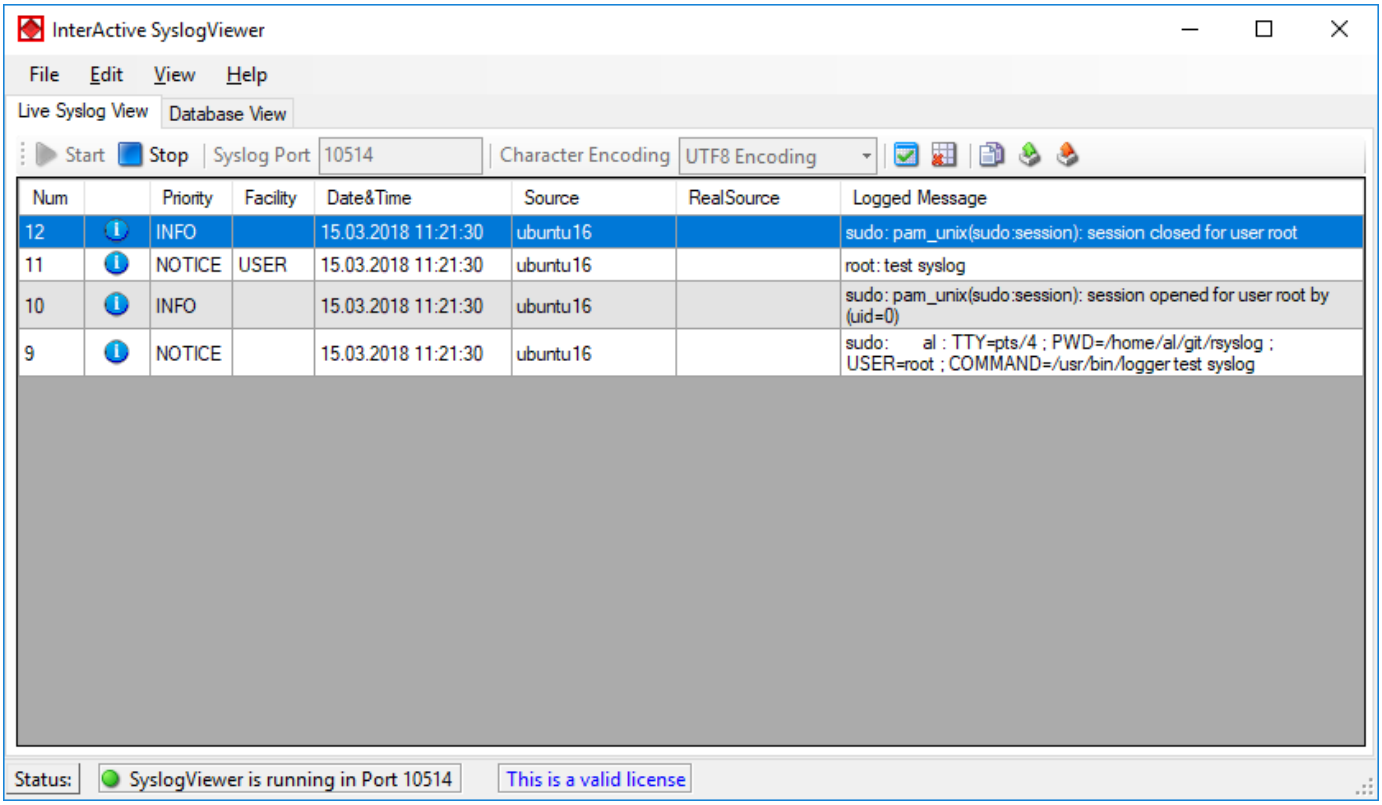
**About**

The About-window will give you additional information to the tool, like the program version.

## Live Syslog View

Primarily, the InterActive SyslogViewer is used for viewing current syslog traffic. All messages are shown in a list with the most important information. These are the Priority, Facility, Date&Time, Source, RealSource and the Message. At the beginning of each line you can see the number of the logged event and a checkbox, for you to track if a message has been read.

The status bar at the bottom of the screen shows you, if the SyslogViewer is running or stopped and how much time you have left for the trial or your licensing status.




- InterActive SyslogViewer Syslog - Live Syslog View*

The toolbar provides you with direct access to the most important functions. These are described here:

**Start**

With the start button, you start the receiving service. Now the InterActive SyslogViewer will receive and display all incoming messages. If messages were sent before starting the service, they will be dropped.

**Stop**

Here you can stop the receiving server.

**Syslog Port**

Here you can define the syslog port where the Viewer should receive the syslog messages.

**Character Encoding**

Here you can define how characters will be decoded. You can choose from Default Encoding (depending on OS), ASCII, Unicode, UTF8, or UTF32.

**Clear checked**

With this button, you can clear all the checkboxes in front of the messages.

**Clear View**

By clicking on this button, all data will be deleted from your data grid.

**Copy Selection**

This helps you copying the selected messages.

**Export Selection**

You can export the selected data directly by using this button.

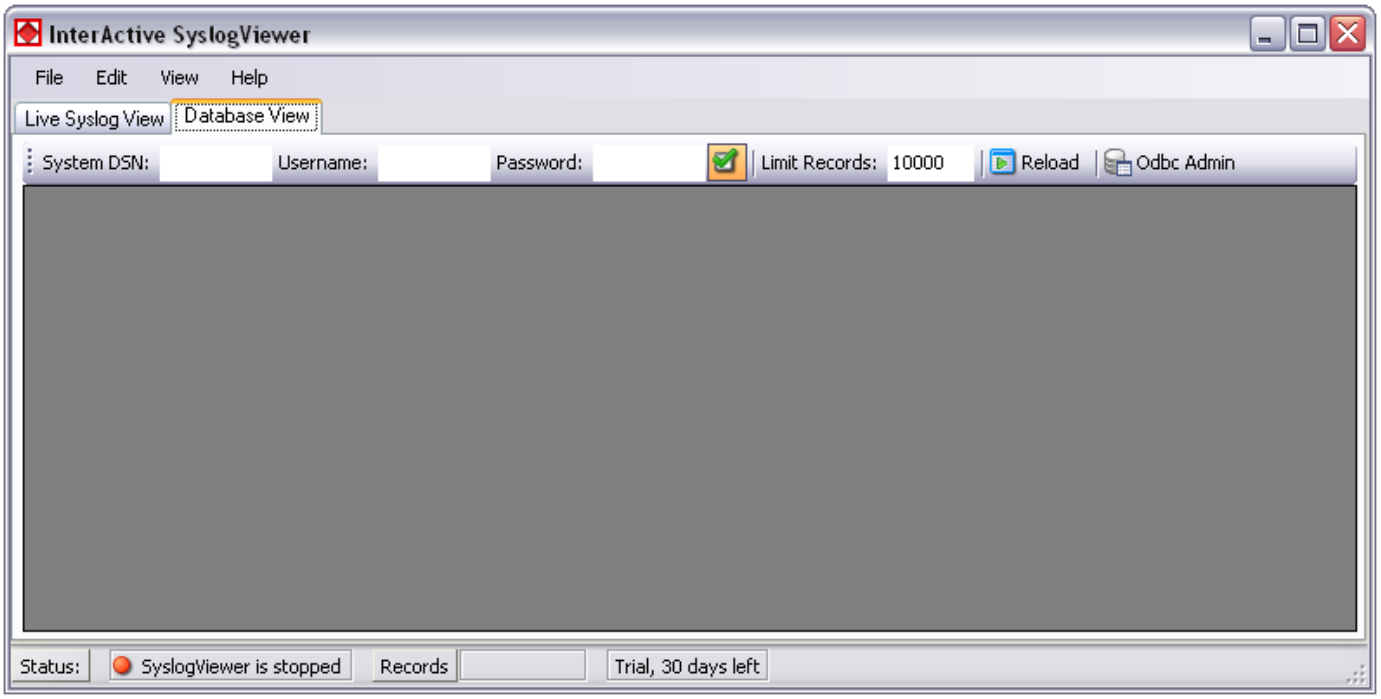
**Export All**

Export the complete data that is in the data grid.

## Database View

Another feature is the possibility to review log messages which are stored in a database.

The status bar at the bottom of the screen shows you, if the SyslogViewer is running or stopped, how many records are currently shown and how much time you have left for the trial or your licensing status.




- InterActive SyslogViewer Syslog - Database View*

The toolbar in this case is for entering the login information for the database.

**System DSN**

Specify the System DSN of your database here.

**Username**

The username for the database.

**Password**

The appropriate password for the database.

**Store Username and Password**

With the checkbox you can tell the InterActive SyslogViewer to keep the username and password or not. This is to make usage easier for you.

**Limit Records**

This limits the maximum of the shown records. The default value is 10000. If changed, this can have a enormous impact on your machine.

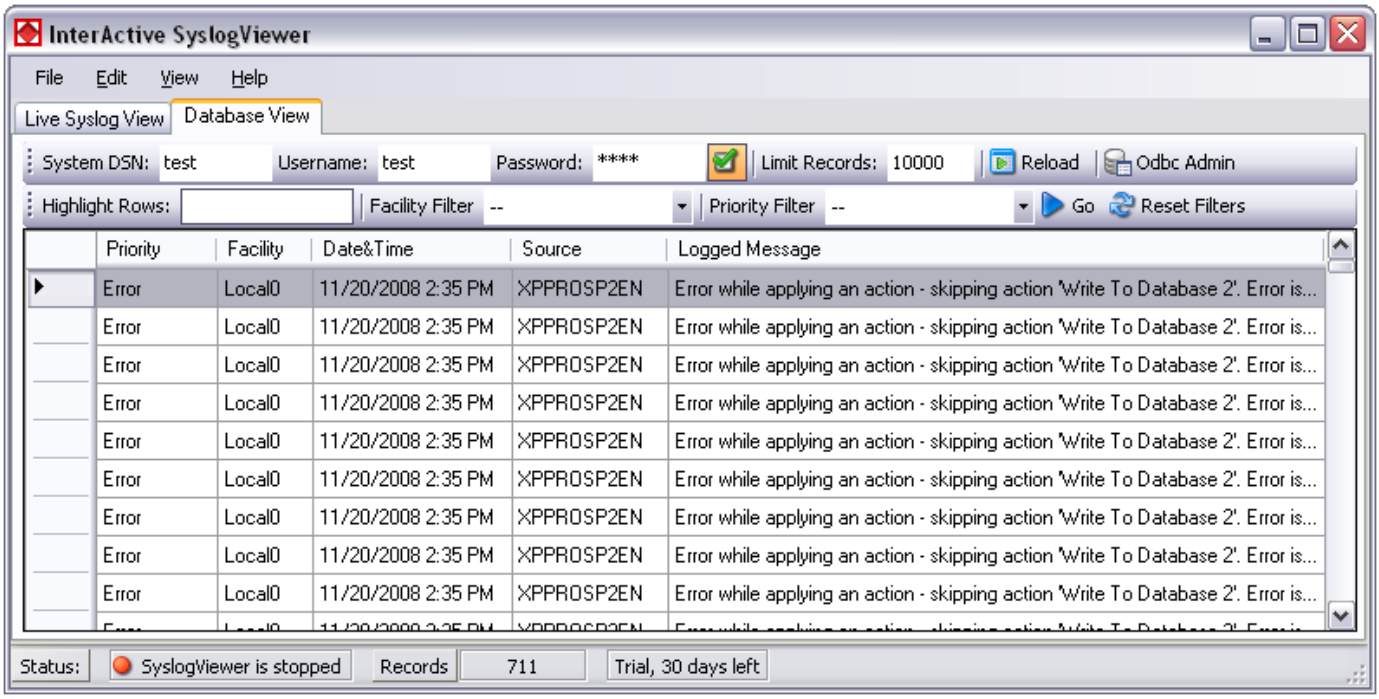
**Reload**

This button is to reload the database. This is needed to view if there are new log messages in the database.

**Odbc Admin**

This button opens the Administration Panel for ODBC Data Source connections

Once a database connection is successfully established, you can see another toolbar with the filter options:




- InterActive SyslogViewer Syslog - Active Database View*

**Highlight Rows**

You can enter a keyword into the field, the rows containing this keyword will be highlighted. You can then find the messages much easier,

**Facility Filter**

Allows you to only show messages with a certain facility. You can use the drop-down menu to specify the facility.

**Priority Filter**

Allows you to only show messages with a certain priority. You can use the drop-down menu to specify the priority.

**Go**

With this button, you apply the filter settings to the current view. Depending on the filter settings you chose you will see either colored lines and/or only the lines from the category you wish to see.

**Reset Filters**

Resets the filter settings and returns you to the default view of your database.

## FAQ

Here you find FAQ about Interactive Syslog Viewer:

# How to Autostart Interactive Syslog Viewer

## Overview

This FAQ explains how to configure Interactive Syslog Viewer to start automatically when Windows boots, ensuring continuous syslog monitoring without manual intervention.

## Background

Interactive Syslog Viewer is a Windows application designed for real-time syslog message monitoring and analysis. By default, it must be launched manually after each system startup. For production environments or dedicated monitoring systems, automatic startup provides continuous availability.

The application supports command-line parameters that enable auto-listening mode, making it suitable for unattended operation.

## Methods for Autostarting Interactive Syslog Viewer

There are two primary methods to configure Interactive Syslog Viewer to start automatically on Windows:

### Method 1: Using Windows Startup Folder (Recommended for Current User)

This method starts the application automatically when the current user logs in.

**Steps:**

1. **Locate the Startup folder**
   - Press `Win + R` to open the Run dialog
   - Type `shell:startup` and press Enter
   - This opens the Startup folder for the current user
2. **Create a shortcut to Interactive Syslog Viewer**
   - Navigate to the Interactive Syslog Viewer installation directory (typically `C:\Program Files (x86)\WinSyslog\` or `C:\Program Files (x86)\WinSyslog\`)
   - Right-click on `InterActive SyslogViewer.exe`
   - Select “Create shortcut”
   - A shortcut will be created in the same directory
3. **Configure the shortcut with command-line parameters**
   - Right-click the newly created shortcut and select “Properties”
   - In the “Target” field, add the `/autolisten` parameter after the executable path
   - Example: `"C:\Program Files (x86)\WinSyslog\InterActive SyslogViewer.exe" /autolisten`
   - Optionally, add other parameters such as `/port=10514` to customize the listening port
   - Click “OK” to save changes
4. **Move the shortcut to the Startup folder**
   - Cut the configured shortcut from the installation directory
   - Paste it into the Startup folder opened in step 1
5. **Verify the configuration**
   - Restart your computer or log off and log back in
   - Interactive Syslog Viewer should start automatically and begin listening for syslog messages

### Method 2: Using Windows Task Scheduler (Recommended for System-Wide Startup)

This method starts the application at system boot, regardless of user login, making it suitable for dedicated monitoring servers.

**Steps:**

1. **Open Task Scheduler**
   - Press `Win + R` to open the Run dialog

- Type `taskschd.msc` and press Enter
- Task Scheduler opens

2. **Create a new task**
   - In the right pane, click “Create Task…” (not “Create Basic Task”)
   - This opens the Create Task dialog
3. **Configure General settings**
   - **Name:** Enter a descriptive name, such as “Interactive Syslog Viewer Autostart”
   - **Description:** Optional description, e.g., “Automatically starts Interactive Syslog Viewer at system boot”
   - **Security options:**
     - Select “Run whether user is logged on or not”
     - Check “Run with highest privileges” if the application requires elevated permissions
     - Choose the user account under which the task should run
   - **Configure for:** Select your Windows version from the dropdown
4. **Configure Triggers**
   - Switch to the “Triggers” tab
   - Click “New…” to create a new trigger
   - In “Begin the task:” dropdown, select “At startup”
   - Optionally, set a delay (e.g., 30 seconds) to allow other services to start first
   - Click “OK” to save the trigger
5. **Configure Actions**
   - Switch to the “Actions” tab
   - Click “New…” to create a new action
   - **Action:** Ensure “Start a program” is selected
   - **Program/script:** Enter the full path to Interactive Syslog Viewer executable

     Example: `C:\Program Files (x86)\WinSyslog\InterActive SyslogViewer.exe`
   - **Add arguments (optional):** Enter command-line parameters

     Example: `/autolisten /port=10514`
   - Click “OK” to save the action
6. **Configure Conditions (Optional)**
   - Switch to the “Conditions” tab
   - Review power and network conditions as needed
   - For dedicated monitoring servers, consider unchecking “Start the task only if the computer is on AC power”
7. **Configure Settings**
   - Switch to the “Settings” tab
   - Recommended settings:
     - Check “Allow task to be run on demand”
     - Check “Run task as soon as possible after a scheduled start is missed”
     - Uncheck “Stop the task if it runs longer than” (to allow continuous operation)
   - Click “OK” to create the task
8. **Test the configuration**
   - In Task Scheduler, find your newly created task
   - Right-click the task and select “Run”
   - Verify that Interactive Syslog Viewer starts correctly
   - Restart your computer to confirm automatic startup

### Available Command-Line Parameters

Interactive Syslog Viewer supports the following command-line parameters for customization:

- `/autolisten` - Automatically start the syslog server upon launch
- `/port=<port_number>` - Override the configured listening port (e.g., `/port=10514`)

- `/windowpos <x>,<y>,<width>,<height>` - Set the default window position and size (e.g., `/windowpos 0,0,512,800`)
- `/?` - Display available command-line options

Example combined usage:

```
"C:\Program Files (x86)\WinSyslog\InterActive SyslogViewer.exe" /autolisten /port=514 /windowpos 0,0,1024,768
```

### Summary

Interactive Syslog Viewer can be configured to start automatically on Windows using either the Startup folder (for user-based startup) or Task Scheduler (for system-wide startup). The `/autolisten` command-line parameter enables automatic syslog listening, making the application suitable for unattended monitoring scenarios.

For production environments requiring robust syslog collection, logging, and processing capabilities, consider using WinSyslog or WinSyslog, which are designed specifically for server-side syslog operations.

further FAQ you find on adiscon.com :

- FAQ

# Configuring

WinSyslog is easy to use and is powerful.

In this chapter, you will learn how to configure the WinSyslog service.

## Core concepts

Use this section to understand how WinSyslog thinks about incoming events and how configuration objects interact. These pages are the conceptual backbone for input services, rulesets, filters, and actions.

If you are new to the product, read this section after `Getting Started` and before deep configuration work.

### Concept map

WinSyslog processing follows this model:

1. An **input service** receives or generates an event.
2. The event becomes an **information unit** inside WinSyslog.
3. The **rule engine** evaluates the event against **rules** and **filter conditions**.
4. Matching **actions** store, forward, display, or transform the event.

In plain language, you can read this as:

```
input service -> ruleset -> action
```

### Canonical concept pages

## WinSyslog - Services

Services inside WinSyslog gather or generate the data that is processed by rules. In plain language, they are the configured inputs and generators inside the product. For example, the `Syslog server` service receives incoming syslog messages and the Windows Event Log Monitor service reads Windows Event Log entries.

In this manual, **service** is the main operational term and **input** is the plain-language concept for the receive side. Some GUI pages still use exact labels such as `Syslog server`, `RELP Listener`, and `SETP Server` for individual service types. Use What do “service”, “input”, “listener”, and “server” mean in WinSyslog? for the terminology mapping.

Typically, there can be multiple instances of the same service running, as long as their listen settings do not conflict. For example, there can be multiple syslog input services on a given system as long as they use different port, address, or transport combinations. For example, there could be three of them: two on the default port of 514, one with TCP and one with UDP, and a third on UDP port 10514. All three can coexist and run at the same time.

**The following services are supported:**

**Heartbeat**

This service generates a special information type. Its primary purpose is to notify a receiving system that WinSyslog, set for heart beating is still alive. So the receiving system can be configured to raise alarms (or corrective actions) if it does not receive heartbeats from WinSyslog.

**MonitorWare Echo Reply**

A central agent running WinSyslog is using the echo request and instructs to poll each of the other WinSyslog services. When the request is not carried out successfully, an alert is generated. The MonitorWare echo protocol

**RELP Listener**

The `RELP Listener` service receives messages over the reliable event logging protocol (RELP). In practice, it is a network input service for RELP traffic. It automatically listens on all available IP addresses, including IPv4 and IPv6, due to the librelp implementation.

Apart from using a different application protocol over TCP, the `RELP Listener` service plays the same operational role as the `Syslog server` service: it receives incoming events and passes them into a ruleset.

**SETP server**

Implements the `SETP Server` service. It is used to receive event notifications reliably from compatible Adiscon products and pass them into a ruleset. In plain language, it is a SETP input service.

**SNMP Trap Receiver**

The `SNMP Trap Receiver` service allows you to receive SNMP trap messages. Roughly speaking, a trap is similar to a syslog message, but it is sent over SNMP. It is generated by the device when the device decides it should send a notification, and it includes the information the device wants to transmit, plus a small set of standard SNMP fields such as version and community.

**Syslog server**

Implements the `Syslog server` service. It receives syslog messages and can listen on any valid port. UDP and TCP communication are supported.

**Event Log Monitor**

Monitors Windows event logs. As soon as new events are detected, these are forwarded to WinSyslog processing. This service is similar to the Adiscon EventReporter functionality.

**Associated rulesets**

Each instance of a service has an associated ruleset. This allows easy creation of customized rulesets on a per-service basis. Of course, multiple services can also operate on a common ruleset.

All services are executed as multiple threads inside WinSyslog. From the operating point of view, there is only one system service called the “WinSyslog”. If the service configuration of WinSyslog is modified, the WinSyslog service needs to be restarted in order to activate the new configuration. Later releases will have some options to automate this task.

### Information Units

Information units contain the data gathered by the services. As soon as a service detects a reportable event, it creates a new information unit. The information unit contains a textual representation of the event (for example a syslog message) as well as information about the event itself. For example, it contains the system that the event was originated from and the date and time it was received.

Which data is contained in the information unit depends on its type. However, there are a number of common data elements present in all information units. Most of these elements can be used as filter conditions in the rule engine. Information unit specific data elements are not eligible as filter conditions. However, there are data elements (properties) which are defined to be present in all information units even though they seem to be specific to a service type. One example is syslog priority. These values are present in each information unit type simply because priority is a good abstraction for other types, too. Such generally available properties are mapped if they are not directly supported by the service type. In the example, an Event Log Monitor maps the event log severity to the syslog priority.

There is a direct one-to-one relation between service type and information unit type. Each service type has its own information unit type.

Inside the rule base, the information unit type itself can be used as a filter condition. This facilitates creating rules that check information unit type specific properties only if they originated from the specific service type (e.g. check syslog priority only if the information unit was generated by a Syslog server).

## Rules

Rules are the workhorse of the WinSyslog. All actions and processing carried out is configured by the rules defined. Rules are configured by the client and processed by the so-called “rule engine” inside the WinSyslog service.

You might already know something similar to the WinSyslog rule engine. Rule engines and rule bases are an extremely powerful tool and in widespread use in the industry. Examples of rule bases can be found at Checkpoint’s Firewall One Firewall Rule Base or Cisco Routing filter - just to name a few.

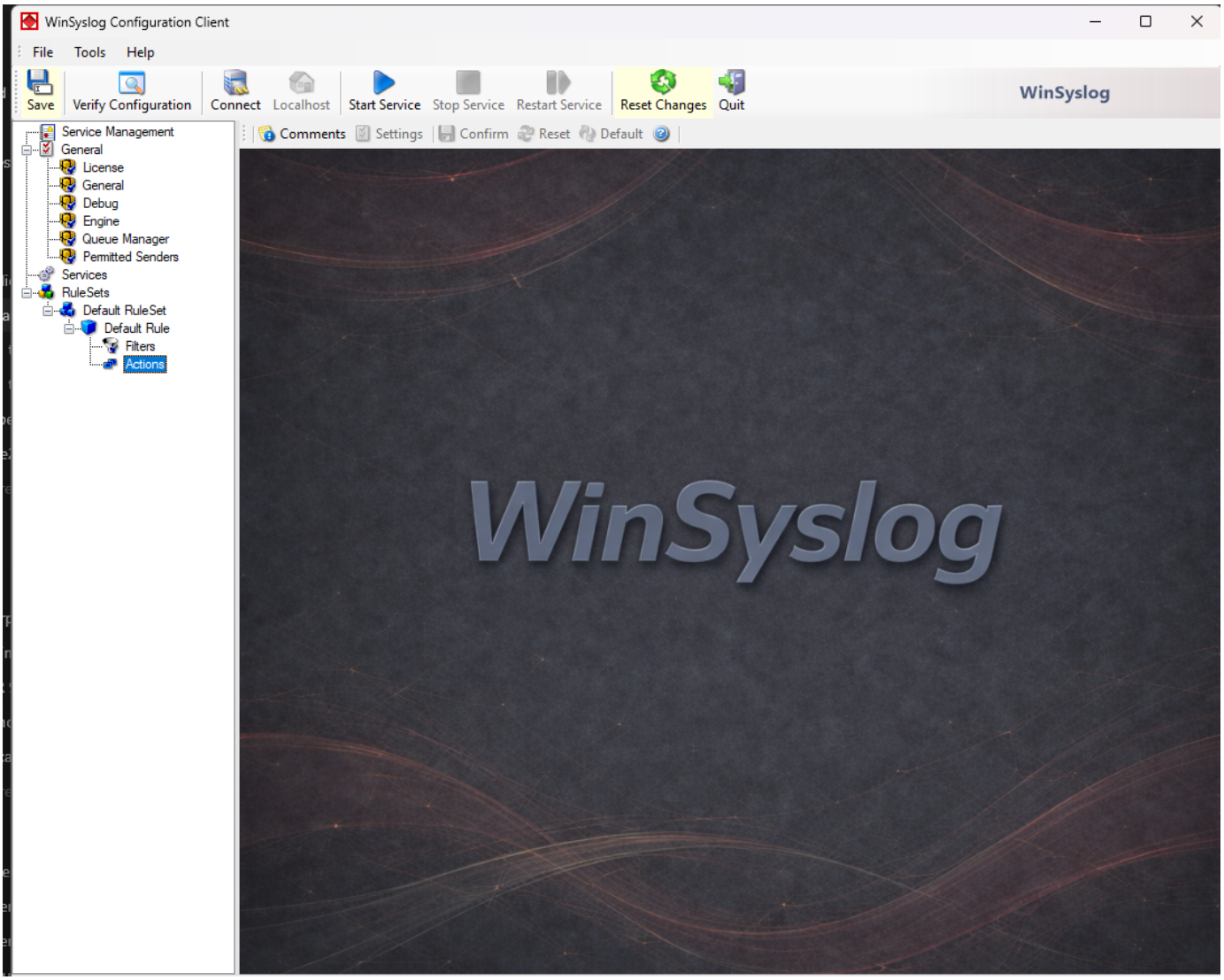
The rule base consists of the rules as configured in the client. The rule engine is the process carrying out the rules. A rule base can contain no, one or an unlimited number or rules. However, if there is no rule at all defined, no action will ever be carried out by the agent. Consequently, the client will issue a warning message in this case.

A rule has a description, associated match conditions, and actions. The match conditions are called “filter conditions”. These specify when a rule is to be carried out. Again, there can be no, one, or many filter conditions for a single rule. If there are no filter conditions, the rule will always match. This is useful in many cases. If there is more than one filter condition, all filter conditions need to match in order for the rule to match (logical AND).

Actions associated with a rule specify what to do when the associated rule matches (and only the associated rule). Actions carry out the actual processing of messages. For example, actions include logging a message to a flat file or database, sending it via email or forwarding it to syslog daemon or another WinSyslog. There can be no, one, or an unlimited number of actions associated with a rule. However, if no action is associated, the rule will not have any effect. Consequently, the client will issue a warning when writing the rule base. Rules without actions can be useful to temporarily disable a rule with complex filter condition. If there are multiple actions, they are not guaranteed to be carried out in any specific order. If you definitely need an action to be carried out before another one, you currently need to define two rules.

Actions can be modified with action modifiers. These are the strings attached to a specific action. Action modifiers allow customizing a specific behavior of this action. It modifies only this action and only this one, other actions of the same type are not affected - regardless if the appear in the same rule or a very different one. The use of the action modifier depends on the type of action. For example, with syslog forwarding it is the host the syslog message is to be forwarded to. With ODBC database logging it is the DSN and so on. If there is no action modifier, the values configured in the client’s configuration tabs will be used. They are also used for all values that cannot be modified via the action modifier (e.g. the SMTP server address for email forwarding).

Below find a screenshot of a rule base with a number of rules, filter conditions and action modifiers:




*WinSyslog sample rule base*

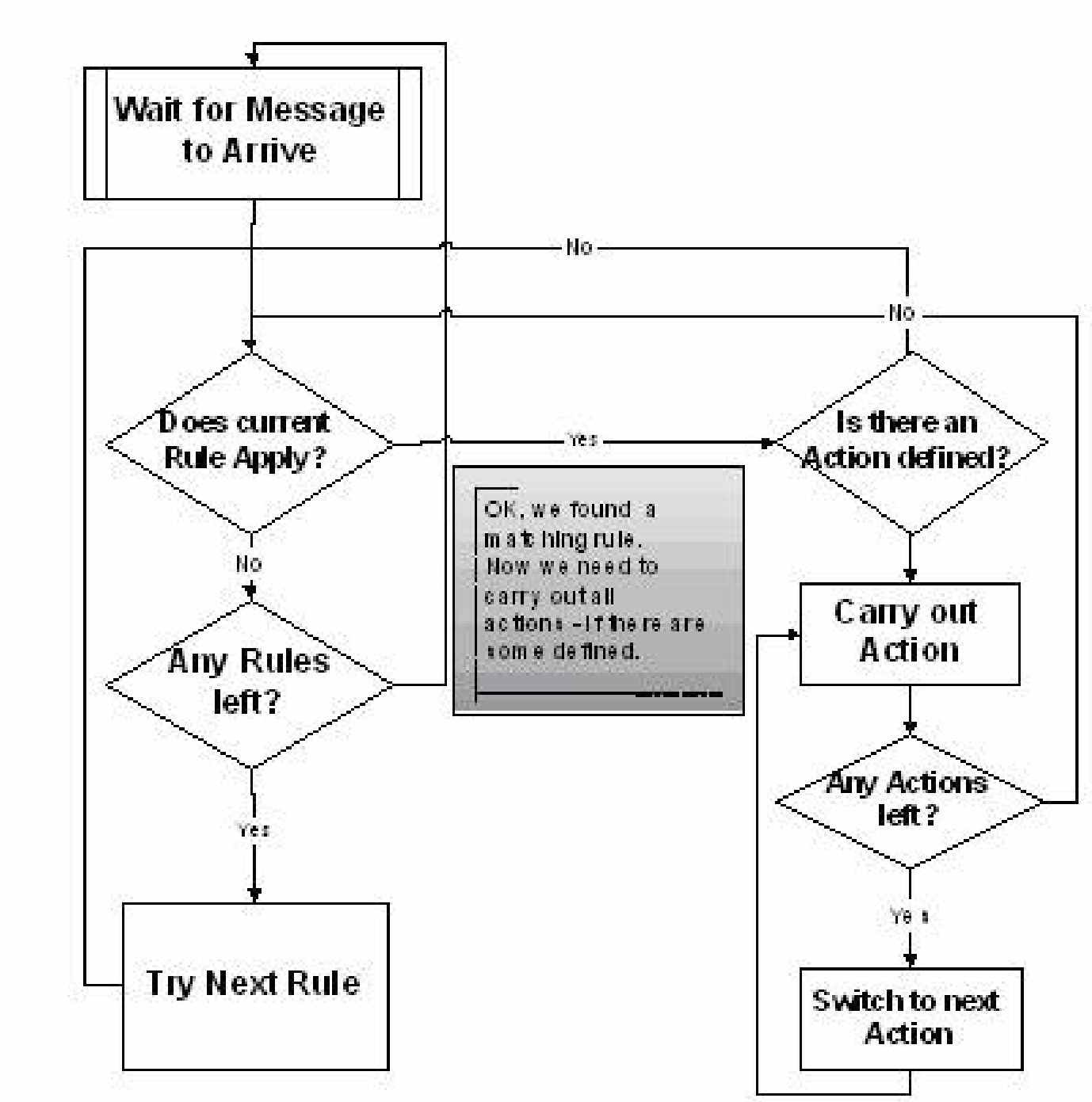
Now that we know the elements, how are rules being processed. It is easy. Rules are strictly processed from top to bottom, or from number one to the last one. For each rule the filter conditions are checked to see if they match. If they do, all associated actions are carried out. Then, the rule engine advances to the next configured rule. Once again, it checks if it matches and - if it does - carries out the actions associated with that rule. Then the next rule is processed and so on. The rule engine stops when there are no more rules to be evaluated. It also stops if a rule contains a “discard” action.

The “discard action” is a very special and powerful action. It does not actually carry out any processing. In fact, it disables all further processing for a message as soon as it is found by the rule engine. So what is the discard action good for? It is used to handle common situations where a number of well know messages - unimportant messages - should be filtered out so that the other rules do not need to take care of these messages. In many other products using rules bases, this is called the “block rule”. Please note that with Adiscon’s rule engine, there can be multiple block rules at multiple layers of the rule base giving you additional flexibility.

One last thing to mention: the rule base is applied to every message arriving at the WinSyslog. By design, there is no way to modify the behavior of the rule base for the next message to be arrived. This ensures an always consistent processing of incoming messages. However, there can be multiple rule bases. Each rule base is associated with a service. Only the rule base associated with the service generating the message will be processed.

While building and testing your rule base, please keep in mind that the WinSyslog service needs to be restarted to load a modified rule base. The reason is that the service does not re-read the rule base to save system resources.

There is an online seminar available on the rule engine and its processing. If you are interested in a more in-depth view, you might want to visit it at rule engine.




Rule Engine Flowchart

For those interested in more in-depth information on how the rule engine works, this flowchart might be helpful:

There is an online seminar available on the rule engine and its processing. If you are interested in a more in-depth view, you might want to visit it at rule engine.

## The Rule Engine

### Overview

This paper explains you the Rule Engine that is employed in some of the MonitorWare Line of Products namely WinSyslog, WinSyslog, and EventReporter 6.0 (and higher)

### What is the Rule Engine

Rule Engine is actually an engine present in the above mentioned MonitorWare Line of Products using which you can define certain filters and the actions that are to be carried out if the defined filter condition matches with the real time condition.

Rule Engine revolves around four basic concepts:

- Information Unit
- Information Services
- RuleSets
- Queue Manager

In order to understand the complete Rule Engine, you need to understand the above mentioned four concepts. The details of these are written below

### 1. Information Unit (Info Unit)

“Information Unit” or “Info Unit”, as we call it, is the basic building block of Rule Engine. Info Unit is basically an object that contains all the information about a specific event which includes:

- Message
- Which application generated this event
- When this event was generated
- Syslog Facility
- Syslog Priority
- Info Unit Type (it tells which Info Service has generated this Info Unit)
- etc

The following figure will give you an idea about an Info Unit:

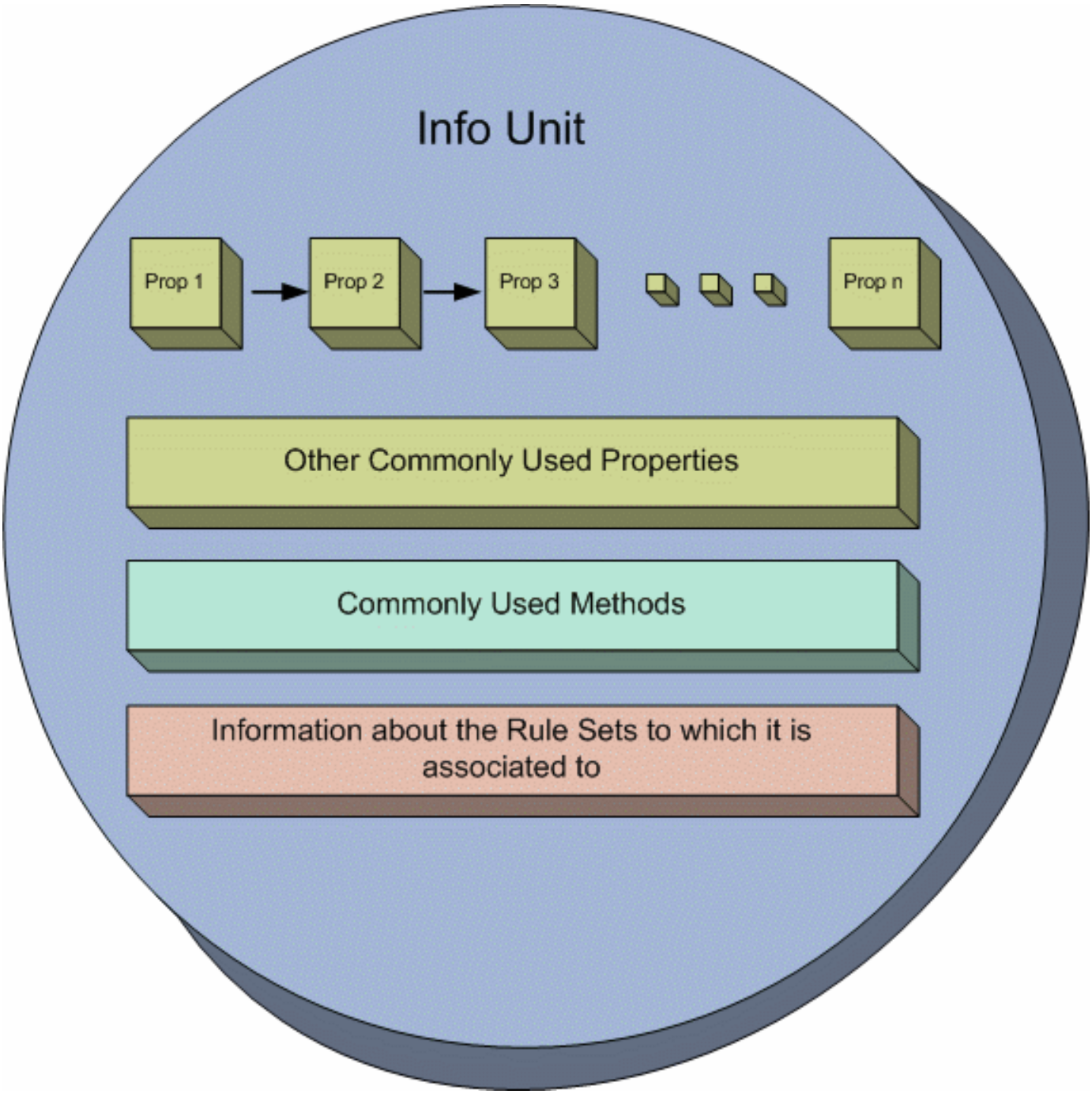



Figure 1: Conceptual Diagram of an Info Unit

As the figure illustrates, an Info Unit contains most of the properties (mentioned above in bullets) in the form of a list. In addition to this list, it also has some commonly used properties separately stored in it (for efficiency reasons). Apart from the properties, an Info Unit also has some methods which allow it to write it or to construct itself etc. Information about the RuleSets (will be explained in the coming sections) is also contained within each Info Unit so that it exactly knows which rules will be applied on it.

### 2. Information Services (Info Service)

“Information Services” or “Info Services”, as we call them, generate Info Units. Each Information Service will generate its own Info Units. The important thing to note over here is that each Info Unit has the same format but can have different properties and rulesets associated with it. For example, if an architect makes a building plan then it becomes a template. Now he can use this template to construct as many buildings as he likes but each one can have different properties (they can differ in color scheme, window styles etc). Exactly in the similar way, an Info Unit is actually a template from which each Info Service makes a specific object of Info Unit that might differ in properties from another Info Unit object.

### Examples of Info Services

There can be a number of different examples on Info Services. Following are some of the examples:

**1. Syslog server**

It receives the messages that are forwarded to it and for each message (or event) it generates an Info Unit out of it.

**2. Event Log Monitor**

It picks up the events from the Window's Event Log and for each event it constructs an Info Unit.

**3. Ping Probe**

It pings a specified device and if doesn't find a response from the other side, it generates an Info Unit with desired information.

### Important Note

One thing to note about Info Services is that there can a number of different Services running on the same machine. You can even run the different instances of the same Info Service (but with different properties naturally). In either case, each Info Service will generate its own Info Unit.

### 3. RuleSets

As the name suggests, a RuleSet is a set of Rules. A "Rule" consists of the following two things

- Filter Condition
- Actions

You might have noticed that the point 1 written above is singular and point 2 is plural which clearly means that you can define only one Filter condition for one rule but can define as many actions as you like. The filter condition can however contain as many Boolean operators as you like.

### Filter Condition

Filter Condition is a combination of different Boolean operators which will evaluate to a Boolean answer. In simple words, the result of a filter condition can either be True or False.

### Actions

Actions are all those events which are fired when a filter condition evaluates to a True value. As mentioned above, a Rule can have more than one actions associated with it which means that if a filter condition evaluates to a true value then all of the actions associated with that rule will execute. If the filter condition, on the other hand, evaluates to a false value then all of the defined actions will be skipped.

Note that other than normal actions, there are three special kinds of Actions that are worth mentioning here:

- Discard Action (Explained Later)
- Include Action (Explained Later)
- Actions that can alter the contents of Info Units permanently

### 4. Queue Manager

Queue Manager simply maintains a queue of all of the Info Units that have been forwarded to it by different Info Services.

### Overall Picture

This section will explain you that how the different components are related to each other and how does the whole process work. The picture shown below gives an idea about how things are working. As you can see that we have four different stages through which the events are processed.

Info Services picks up the events and convert them into Info Unit. Note that each Info Service has its own Info Unit. These Info Units are passed to the Queue Manager. The job of the Queue Manager, as mentioned above and as clear from the diagram, is to simply make a queue of these Info Units that it has received from various Info Services. The Rule Engine picks up the Info Units from this Queue Manager, applies the rules on these Info Units (as mentioned above, each Info Unit has the information about which rules should be applied on it) and if necessary

carries on the actions. The rule engine keeps on repeating this process while there are some Info Units present in the Queue.

Manager's Queue

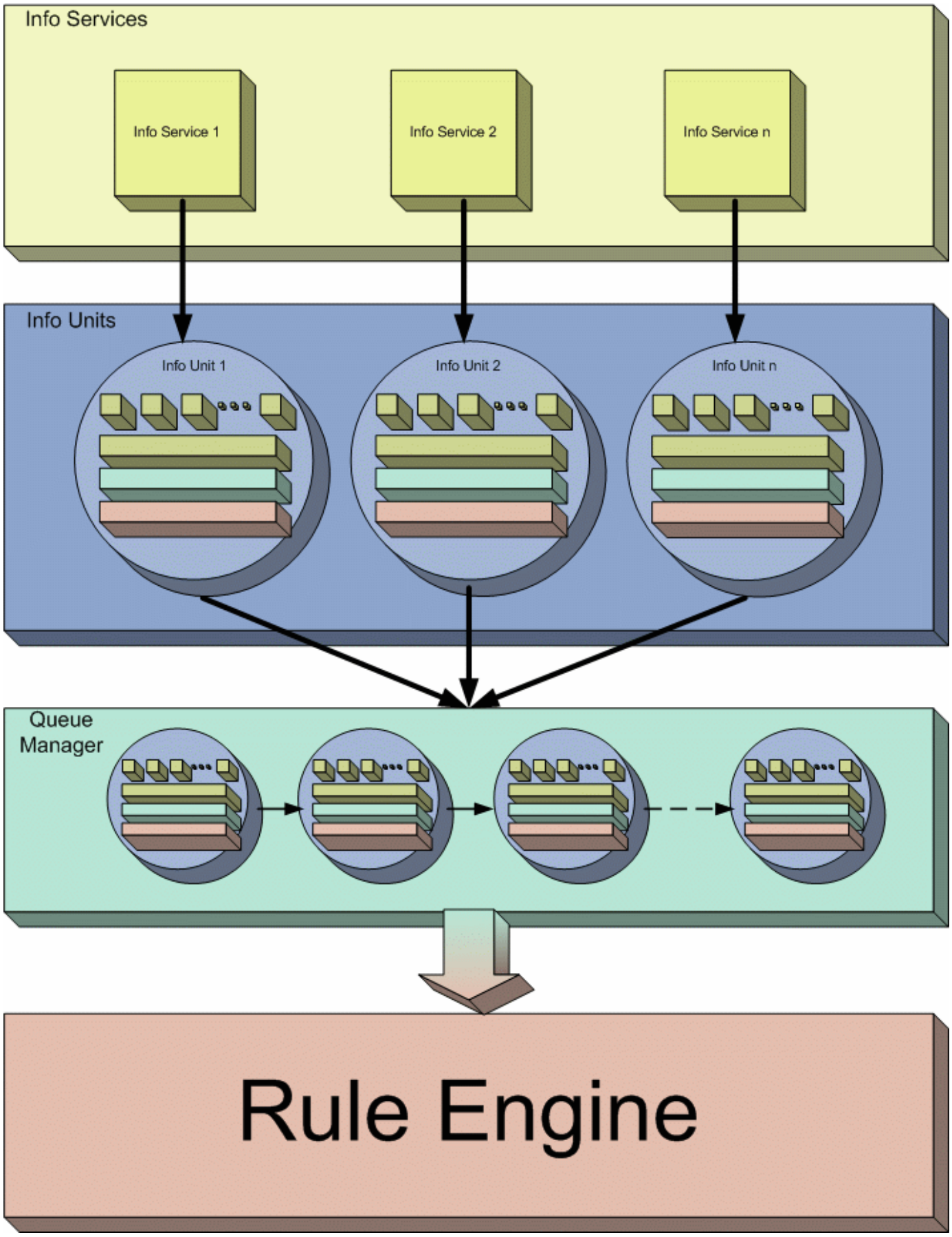



Figure 2: Overall Process

### How Does the Rule Engine Work

Having explained the overall picture of the whole process, let’s specifically talk about Rule Engine. The following figure explains it in detail:

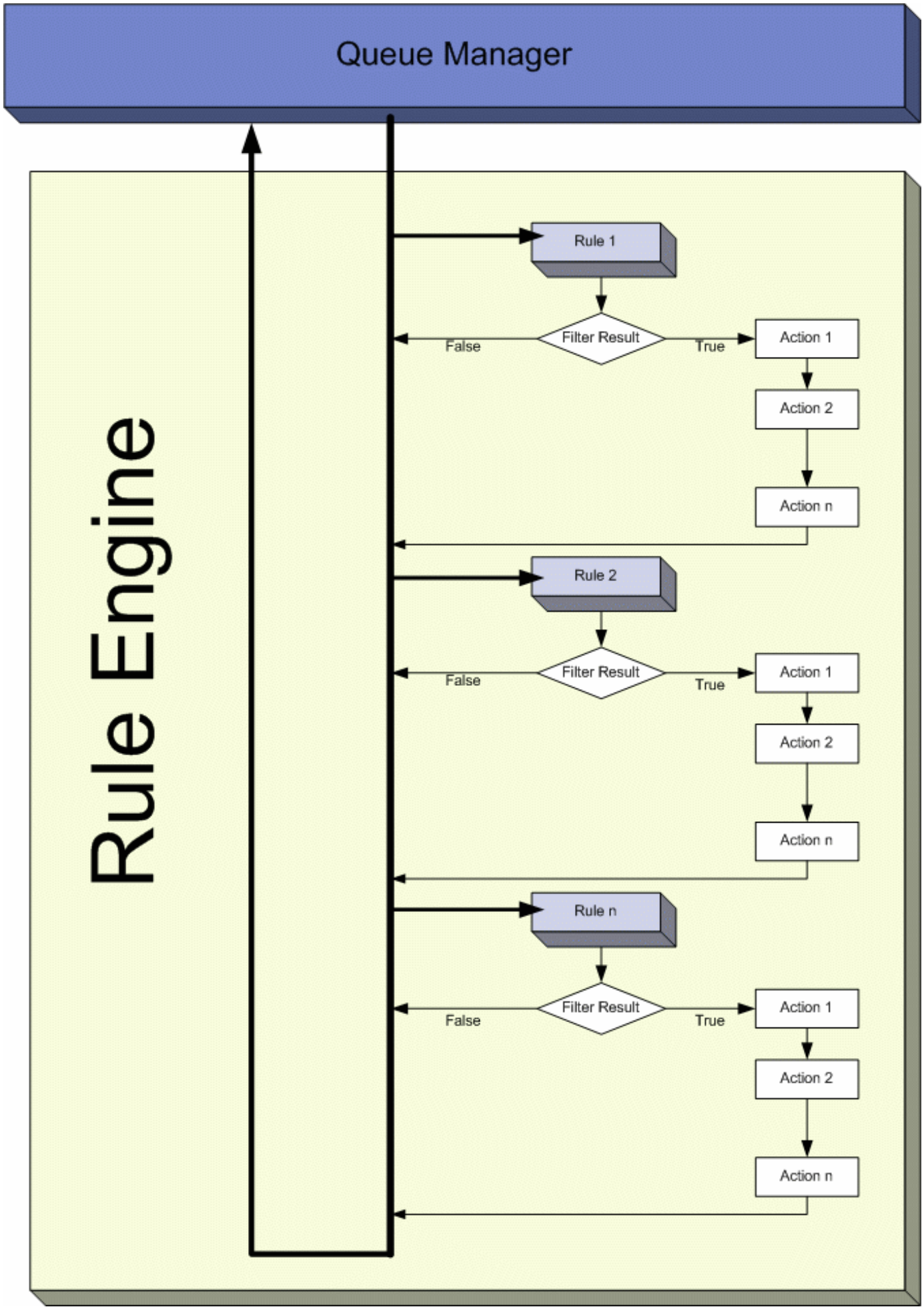

Queue Manager
Rule Engine
Rule 1
Filter Result
False
True
Action 1
Action 2
Action n
Rule 2
Filter Result
False
True
Action 1
Action 2
Action n
Rule n
Filter Result
False
True
Action 1
Action 2
Action n

Figure 3: Working of Rule Engine

As you can see in the figure, the Rule Engine picks up Info Units one by one from the Queue Manager. Since each Info Unit has the information about its Rule sets, it will apply the rules on it in the same order in which they were defined. As you can see above, it will pick up the first Rule and evaluates its Filter Condition. If that Filter Condition is evaluated to false, all the actions associated with that rule will be skipped and it will pick up the second Rule. If the Filter Condition, on the other hand, evaluates to True, it will execute all the actions that are associated with this Rule in the order in which they were defined. After the execution of all these actions, it will pick up the next rule in the current ruleset. Once all the rules have been executed, the current Info Unit (that was handed over to Rule Engine by the Queue Manager) will be destroyed and the Rule Engine will go to the Queue Manager to pick up the next Info Unit if there exists one.

The above picture has been drawn for normal flow of executions. There can be 2 conditions when the flow will not follow the diagram shown above. These conditions arise in response to 2 special kinds of actions that are called Discard Action and Include Action.

### Discard Action

A Discard Action immediately destroys the current Info Unit and any action of any Rule that has been defined after the Discard Action will not be executed at all. Let's take a simple example to clarify it further.

Let's say that Action 2 of Rule 1 in the picture above is a Discard Action. If the Filter Result of Rule 1 is evaluated to true, then Action 1 will be executed. As Action 2 is a Discard Action, immediately the current Info Unit will be destroyed (which means that now the Rule Engine will skip all the Rules and all the actions associated with them) and the Rule Engine will go back to the Queue Manager to pick up the next Info Unit in the Queue.

### Include Action

An Include Action simply includes another RuleSet in some existing RuleSet. When this Action is encountered, the Rule Engine leaves the normal flow and go to the included ruleset (which may contain many rules as well). It executes all the rules that have been defined in that included RuleSet. After the execution of all of them, it will return to its point from where it left the original flow. Let's take an example to clarify it a little further.

Let's say that the Action 1 or Rule 1 is an include action. If the Filter Condition result of Rule 1 evaluates to true, it will execute the Action 1. Since Action 1 is the include action in this example, it will go to the included ruleset and will execute its Filter Condition. If that filter condition evaluates to true, it will execute all of its actions and will return to Action 2 of Rule 1 (of normal flow) and if on the other hand, the filter condition of the included ruleset evaluates to false, it will skip all of its actions and will come back to the Action 2 of Rule 1 (of normal flow).

Note that there is no limit on including the rules which means that a rule that has been included in another rule may contain another rule in it which might contain another rule in it and so on.

### Suggestions for Defining Complex RuleSets

While defining a complex RuleSet, it might be a good idea to follow the stages defined below.

Edit Stage # - Actions Stage 0 - Discard unwanted events Stage 1 - Post Process Stage 2 - Discard unwanted events Stage 3 - All Actions Stage 4 - Individual Actions

As mentioned above, the rules and actions will be executed in the order in which you will define them. So it's very important that you define the actions in a way such that you achieve the desired results as well as achieve them with efficiency. For example, if you haven't defined any filter which we call as No Filter (it always evaluates to true) and if the first action that you have defined is the Discard Action, then there is no meaning of defining any action after this first action because the first action will always be executed and it will always discard the complete Info Unit.

Here is the explanation of the above mentioned stages.

### Stage 0

In this stage, you can discard those events that you are not interested in. You can use the Discard Action explained above to discard the events.

### Stage 1

In this stage, we recommend to Post Process the incoming Info Units. Once the Info Unit has been handed over to the Rule Engine from the Queue Manager, you can actually change the contents of the Info Units to make them more meaningful.

### Stage 2

In this stage, you might want to again discard those events that you are not interested in. Simply use the Discard Action.

### Stage 3

In this stage, you will apply the actions that will apply to all of the Info Units coming (to be more specific, you will apply those rules over here for which you have selected “No Filter” as the filter condition.

### Stage 4

In this stage, you will create the rules for which you have specific filter conditions.

To sum it up, we recommend doing most generic things first and least generic things later or in other words, do the generic things first and the specific things later. Note that this section suggests only the typical scenario but it can vary from depending upon the needs. For example, you might want to perform some actions on some specific events after stage 1 and before stage 2.

### Filter Conditions

Filter conditions are used inside the rule engine. They help to decide when a rule is to be carried out. Filter conditions are considered to match of the outcome if the configured comparison operation is “TRUE”. Available filter conditions are listed down below:

- Global Conditions
- General Conditions
- Date / Time
- InformationUnit Type
- Syslog
- SNMP Traps
- Custom Property

### Global Conditions

Global Conditions apply to the rule as whole. They are automatically combined with a logical “AND” with the conditions in the filter tree. These are:

- Treat not found Filters as TRUE*

If a property queried in a filter condition is not present in the event, the respective condition normally returns “FALSE”. However, there might be situations where you would prefer if the rule engine would evaluate this to “TRUE” instead. With this option, you can select the intended behavior. If you check it, conditions with properties not found in the event evaluates to “TRUE.

- Fire only if Event occurs* - This is kind of the opposite of the “Minimum Wait

Time”. Here, multiple events must come in before a rule fires. For example Ping is not a very reliable protocol, so a single ping might be lost. Thus, it may not be the best idea to restart some processes just because a single ping failed. It would be much better to wait for repetitive pings to fail before doing so.

Exactly this the “Fire only if Event occurs” filter condition is made for. It waits until a configured amount of the same events occurs within a period. Only if the count is reached, the filter condition matches and the rule can fire.

- Minimum Wait Time* - This filter condition can be used to prevent rules from

firing to often. For example, a rule might be created to check the status of a port probe event. The port probe probes an smtp server. If the event is fired and the rule detects it, it will spawn a process that tries to restart the service. This process will take some time. Maybe the SMTP gateway need some more time to fully start up so that the port probe might fail again while the problem is already taken care of. The port probe as such will generate an additional event. Setting a minimum wait time will prevent this second port probe event to fire again if it is – let us say – within 5 minutes from the original one. In this case, the minimum wait time is not yet reached and as such, the rule will not match. If, however, the same event is generated 5 hours later (with the mail gateway failing again), the rule will once again fire and corrective action taken.

### Date Conditions

Rule processing can be bound to a specific or the installation date. By default a Rule will always be processed.

### General Filter Conditions

This set includes filters which are related to Non-Event Log specific settings. These are:

- Source System* - This is the system a message is originated from. It can be

used to check for authorized systems to pass messages to WinSyslog.

- Message Content* - The message content filter condition is very powerful. It

evaluates to true if the specified content is found anywhere in the message. As there is implicit wildcarding, there is no need to specify extra wildcards.

- CustomerID* - CustomerID is provided for customer ease. For example if someone

monitors his customer’s server, he can store different CustomerIDs in each agent. This is user configurable.

- SystemID* - SystemID is of type integer and is to be used by our customer. In addition, it is user configurable.
- Status Name and Value* - These filter type corresponds to set status action.

### Date / Time

This filter condition is used to check the time frame and/or day of week in which an event occurred.

- Time* - This filter condition is used to check the period in which an event occurred. For example, a syslog message from a Cisco router saying that it dialed up is normal if it occurs during office hours. If it occurs at night, so, it is an alerting signal and an administrator might receive notification of this event (while he might otherwise decide to discard it). This can be done with the time setting.
- Weekdays* - This is closely equivalent to the time filter condition, except that it is applied on a per-day basis. So it can be used to detect for example events occurring on weekends and act differently on them.

### Information Unit Type

This is based on the type of service that generated the information unit. So with this setting rules can be created that act only on e.g. syslog messages or NT event reports.

### Syslog

Syslog related filters are grouped here:

- Syslog Facility* - For syslog information units, this is the actual syslog facility. If that filter condition is used on non-syslog originated information units, it will be a value mapped on a best effort basis to a syslog facility.
- Syslog Priority* - For syslog information units, this is the actual syslog priority. If that filter condition is used on non-syslog originated information units, it will be a value mapped on a best effort basis to a syslog priority.
- Syslog Tag* - The syslog tag value, is a short string. This is provided for non-syslog messages based on configuration. In most cases, this is used for filtering.

### SNMP Traps

Using SNMP Traps WinSyslog can be used to manage and monitor all sorts of equipment including computers, routers, wiring hubs etc. A trap is generated when the device feels it should do so and it contains the information that the device feels should be transmitted. Related filters are grouped here:

- Community* - It corresponds to the respective SNMP entity.
- Enterprise* - It corresponds to the respective SNMP entity.
- Generic name* - It corresponds to the respective SNMP entity.
- Version* - It corresponds to the respective SNMP entity.
- Uptime* - It corresponds to the respective SNMP entity.

### Custom Property

As the name suggests it is a “Custom Property”. Internally in WinSyslog all values are stored in properties. For example the main message is stored in a property called “msg”. By using this dialog you can access properties which are dynamic (Like those from SNMP Trap Monitor when using v2 protocol).

### Actions

Actions tell the Product (i.e. WinSyslog or EventReporter or WinSyslog or any of the combinations) what to do with a given event. With actions, you can forward events to a mail recipient or Syslog server, store it in a file or database or do many other things with it. There can be multiple actions for each rule. These actions are described in the following section.

#### Write to File

The message is written to a plain text log file.

#### Write to Database

The message will be written to the specified ODBC database. This database format will be used by the MonitorWare Console that becomes available later. Therefore, if you intend to use the console, we recommend adding at least one rule that persists data to the database.

#### Write to EventLog

The message will be written to the application event log. Please note that the agent intentionally does not try to make the message look like it was generated on the local system. This could be very confusing. Instead, it is written inside the message part with standard values for event source and type.

#### Forward via Email

The message will be forwarded via email. Please note that each message will generate one email message. Messages are not combined to fit into a single mail. The Send Mail Action includes a timeout feature (m_nTimeoutValue) that provides control over message delivery timing.

#### Forward via Syslog

The message will be forwarded to a syslog daemon. UDP and TCP forwarding is supported.

#### Forward via SETP

The message will be forwarded via the custom SETP protocol. This is typically used in environments where data from different agents will be consolidated in a central place. SETP allows to transfer all InformationUnits exactly as they are. As such, the central repository can store an exact picture of the whole network.

#### Net Send

> **Warning**
>
> **Removed from configuration client (26.07).** Retained for reference only.

The message was forwarded via the Windows `net send` functionality. The Windows Messenger service is unreliable on modern Windows and is no longer available from the configuration client. Use **Send Email**, **HTTP REST Output**, or other forwarding actions instead.

#### Start Program

The message will be passed to an external process. The command line is specified in the action modifier.

#### Play Sound Action

This action allows you to play a sound file.

#### Send to Communications Port

This action allows you to send a string to an attached communication device, that is it sends a message through a Serial Port. It can send any message to a configured Serial or Printer port.

#### Set Status

This action allows you to create new properties of your own choice in the incoming messages. There is an internal Status List within the product which you can use for more complex filtering. You can set property over the Set Status action and you can add filter for them. They are more or less helpers for building complex rule constructions.

#### Set Property

With the “Set Property” action, some properties of the incoming message can be modified. This is especially useful if an administrator would like to e.g. rename two equally named devices.

#### Call RuleSet

This Action simply calls another RuleSet in some existing RuleSet. When this Action is encountered, the Rule Engine leaves the normal flow and go to the called RuleSet (which may contain many rules as well). It executes all the rules that have been defined in that called RuleSet. After the execution of all of them, it will return to its point from where it left the original flow.

#### Discard

Please see the rules description below for a complete discussion. Effectively, the message will be discarded and any further processing of this information unit be stopped as soon as a “Discard” action is found.

#### Post-Process Event Action

The post process action allows you to re-parse a message after it has been processed e.g. Tab Delimited format. Such re-parsing is useful if you either have a non-standard syslog format or if you would like to extract specific properties from the message.

### Why this matters

Understanding these concepts helps you:

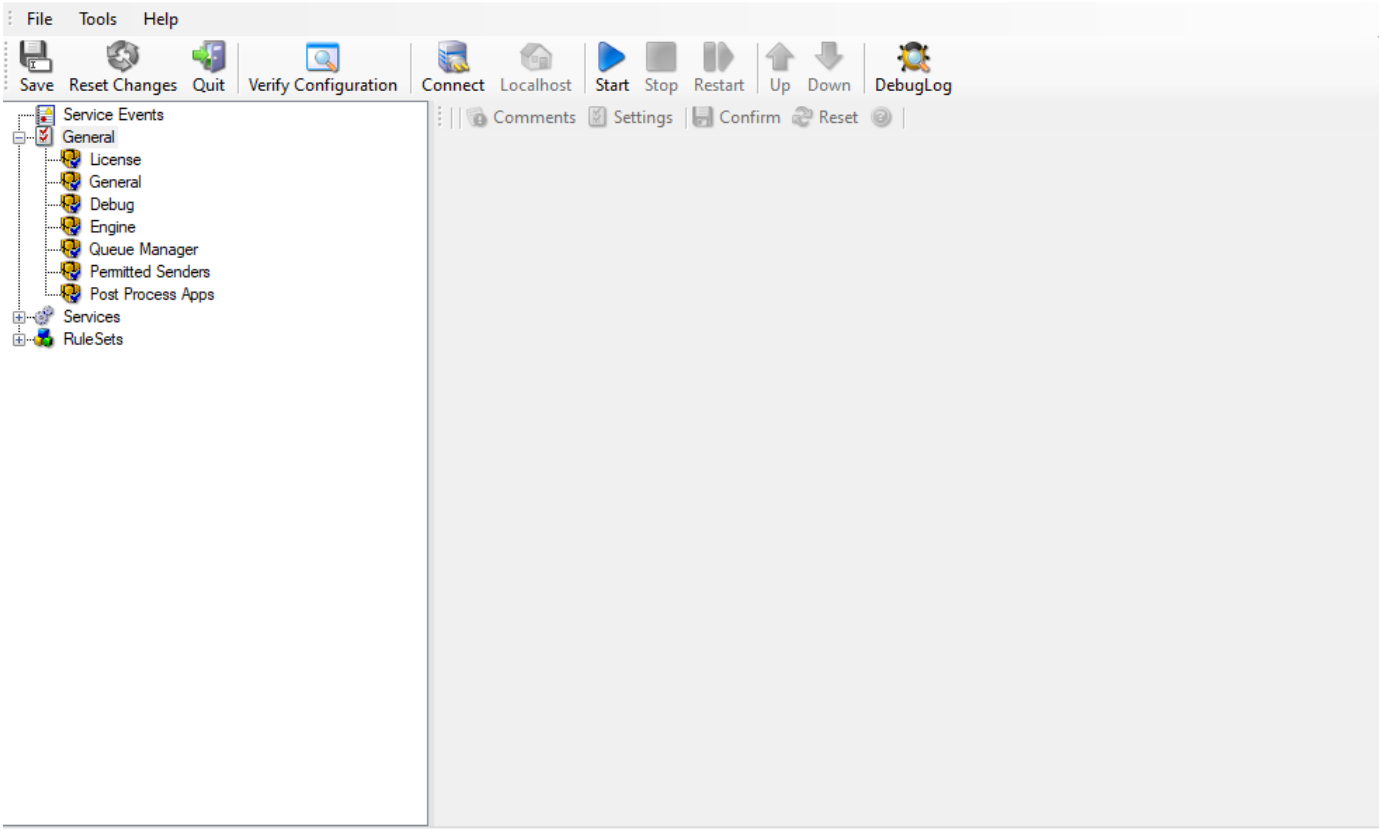
- design rulesets with predictable behavior for each input service
- avoid duplicate or conflicting processing paths
- choose the right action type for storage, forwarding, or alerting
- troubleshoot why an event did or did not match a rule

## Configuring WinSyslog

In this chapter, you will learn how to configure WinSyslog.

The WinSyslog runtime service runs in the background once it is configured. There is no manual intervention needed to operate it. As such, this chapter focuses on the WinSyslog Configuration Client application. It is used to configure input services, rulesets, actions, and related settings.

To run the WinSyslog Configuration Client, simply click its icon present in the WinSyslog program folder located in the Start menu. Once started, a Window similar to the following one appears:




- Configuration Client*

The Configuration Client (“the Client”) has two elements. On the left hand side is a tree view that allows you to select the various elements of the WinSyslog system. On the right hand side are parameters specific to the element selected in the tree view. In the sample above, the right hand side displays the specific parameters for a rule action.

The tree view has three top-level elements: General / Defaults, Running Services, and RuleSets.

Under General / Defaults, basic operational parameters as well as defaults for actions and services are defined. The default themselves do not activate anything. However, the parameters in here are used each time an actual service or action needs a configuration parameter and none is defined in that specific instance. We highly recommend putting the most common parameters into the defaults, which reduces the amount of data entry in the specific elements dramatically. Please note that each default can be overwritten in a specific service or action.

The tree view’s Running Services area lists all configured services as well as their parameters. There is exactly one service entry for each configured input or generator that you create. Please note that there can be as many instances of a specific service type as your application requires. Typically, there can be multiple instances of the same service running, as long as their port, address, and transport settings do not conflict. For example, there can be multiple `Syslog server` services on a given system as long as they listen on different combinations. For example, there could be three of them: two on the default port of 514, one with TCP and one with UDP, and a third on UDP port 10514. All three can coexist and run at the same time. If these services try to use the same effective port, address, and transport combination, Windows logs an error and WinSyslog cannot perform the intended message intake.

In this manual, **input** is the clearest plain-language concept for receive configuration, while **service** remains the main operational term for these configured WinSyslog objects. Some GUI labels still use names such as `Syslog server` and `RELP Listener`. Those are exact current service names in the client.

For the general port, address, and transport conflict rule, including why TCP and TCP+TLS cannot share the same IP address and port, see How Do Port, Address, and Transport Conflicts Work for Input Services?.

Theoretically, you can run a few hundred services in a single service instance. However, both from a usage scenario point of view as well as concerning operating system resources, we recommend limiting the services to a maximum of 20 to 30. Of course, there are some applications where more than this limit is useful. WinSyslog does not restrict this number. If there is a need for a large number of services and the hardware is capable of managing all these tasks, there is nothing in WinSyslog that limits from doing so.

The actual parameters depend on the service type. Common to all services is the capability to enable or disable a service. A service is started only if it is enabled. Otherwise, it will not run, but the configuration data can still be present. That way, it is easy to temporarily disable an input service without deleting it.

Also common to all service types is the association to a ruleset seen at the bottom of the right hand configuration dialog. This specifies which ruleset processes the information units generated by this service.

To create a new service, right click on “Running Services”. Then select “Add Service” and the respective service type from the pop up menu. Then follow the wizard. To delete an existing service, right click it and select “Delete Service”. This removes the configured input or generator and its configuration is now irrecoverable. To temporarily remove a service from operation, simply disable it in the property sheet.

The tree view’s last main element is RuleSets. Here, all rulesets are configured. Directly beneath “Rules” are the individual rulesets. Each set is completely independent of the others. They are just centrally stored so they can be associated with services (see above for an explanation).

Beneath each ruleset are the individual rules. As described in Rules , a rule’s position in the list is vitally important. rules at the top of the rule set are executed before those further down. To move a rule up or down, simply right click it and select “move up” or “move down” from the pop up menu.

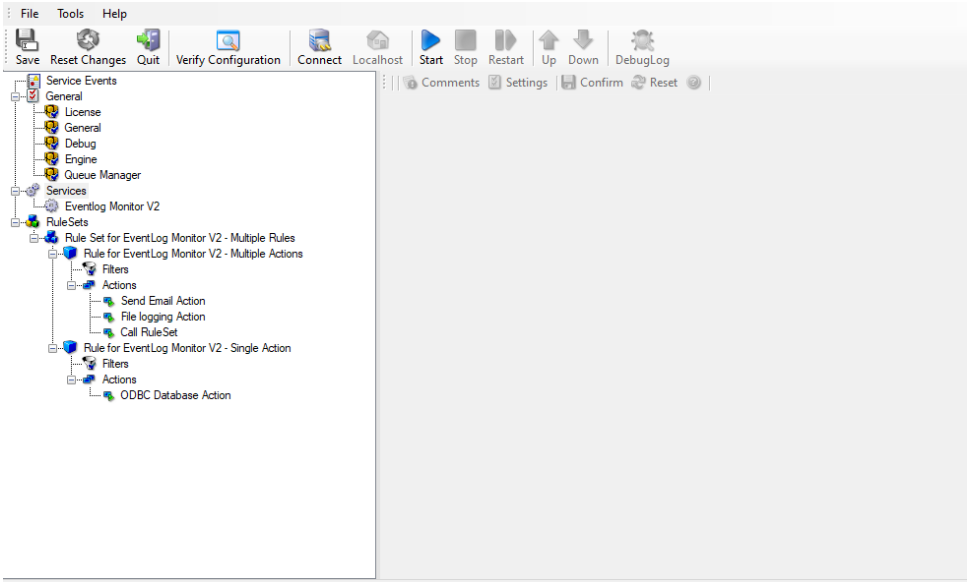
In the tree view, filter conditions and actions are beneath the rule they are associated with. Finally, beneath actions are all actions to carry out.

The following sections describe each element’s properties.

### User Interface changes in 26.07

See User Interface changes for license file UI, YAML import/export, service toolbar changes, and the WinSyslog User Interface.

## Organizing with RuleSets, Rules, and Actions




WinSyslog gives you flexible control over your log data. You can set up as many **RuleSets**, **Rules**, and **Actions** as you need to process your logs the way you want.

**RuleSets** are like folders that help you group your rules. They make it easy to keep your log processing organized. For example, you could:

- Create a separate RuleSet for each input service such as your firewall or web server intake. This keeps your configuration tidy because everything related to a specific input service is in one place.
- Use a single RuleSet for multiple services if your logs require a more general kind of processing.
- Design RuleSets specifically to be called from within other RuleSets. This is done using the “callruleset action” and is useful for reusing logic or structuring complex workflows.

Every RuleSet contains one or more **Rules**. These rules are at the heart of your log processing. They determine precisely what should happen with a log message. Think of a Rule as an “if-then” statement. It has two main parts:

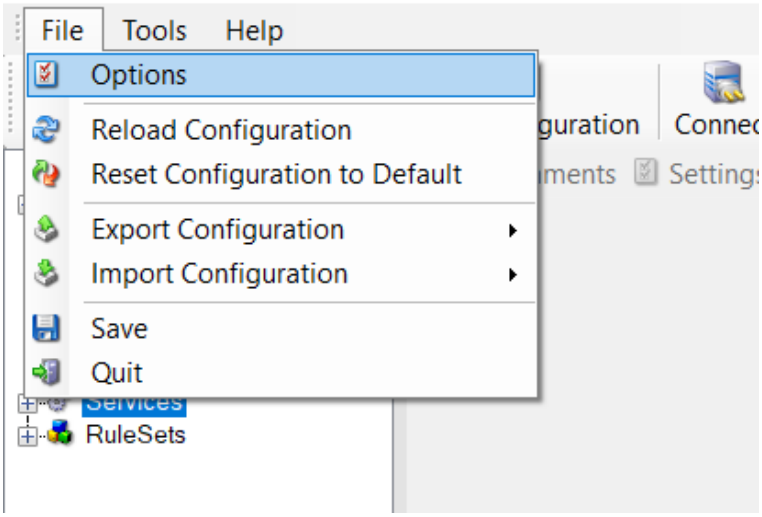
1. **Filter Conditions**: These are the criteria a log message must meet for the rule to apply. For example, “Is it an error message?” or “Did it come from a specific IP address?” You can learn more about them in filter conditions.

2. **Actions**: These are the tasks that will be performed if the filter conditions are met. This could be writing the message to a file, sending it to a database, dispatching an email, and much more. A rule can include one or more actions. If several actions share the same filter conditions, you can conveniently combine them within a single rule.

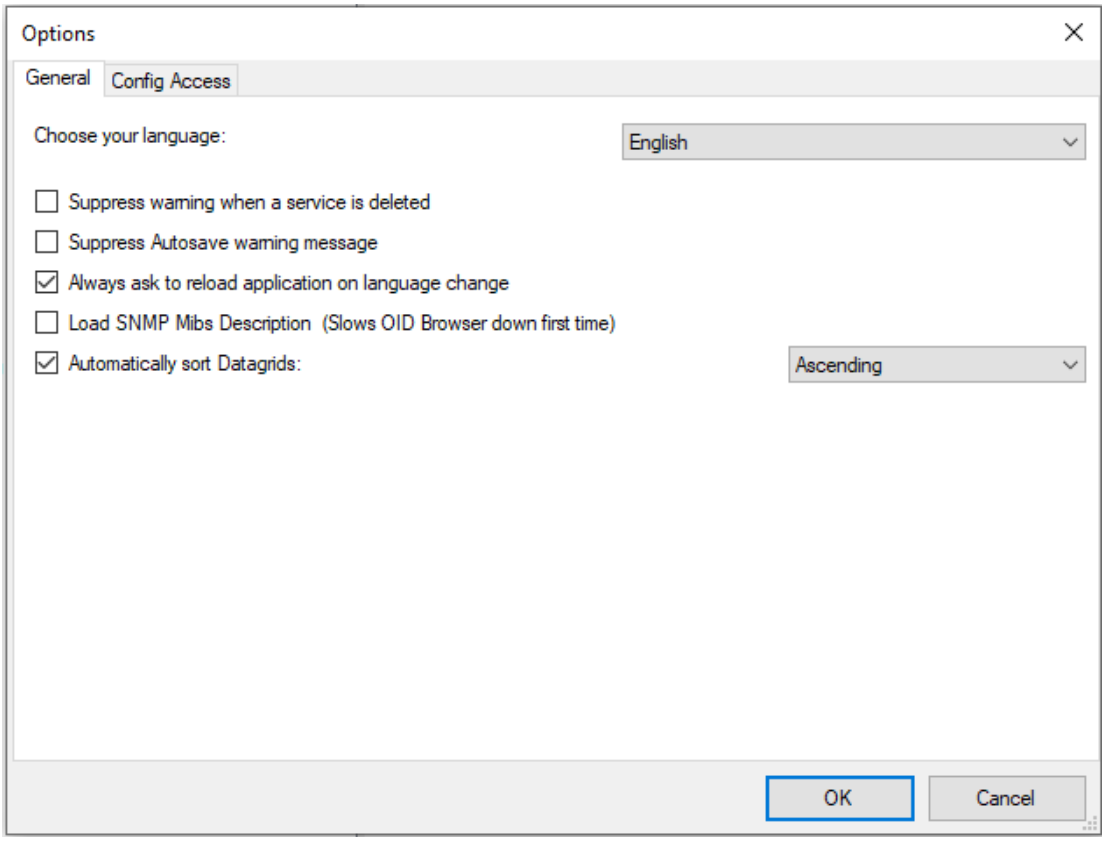
Essentially, with WinSyslog, you use RuleSets to choose the processing area, Rules to define which messages you are interested in, and Actions to specify what WinSyslog should do with them. This structure makes the configuration clearer and helps you find issues faster and implement changes more easily.

## Client Options

There are several options, that refer to the configuration client and not to the service. These can be found under File -> Options




- Client Options*




- General Tab*

**Choose your language**

You can choose a language pack. “English” is the default and suggested language.

**Suppress warning when a service is deleted**

If this option is checked you will not get a warning when you try to delete a service and there is no other service that uses the connected ruleset.

**Suppress autosave warning message**

If you make changes in the configuration and switch to another component, a warning will occur if you haven’t saved the changes. This warning will also allow you to directly enable auto-saving the configuration.

**Always ask to reload application after language change**

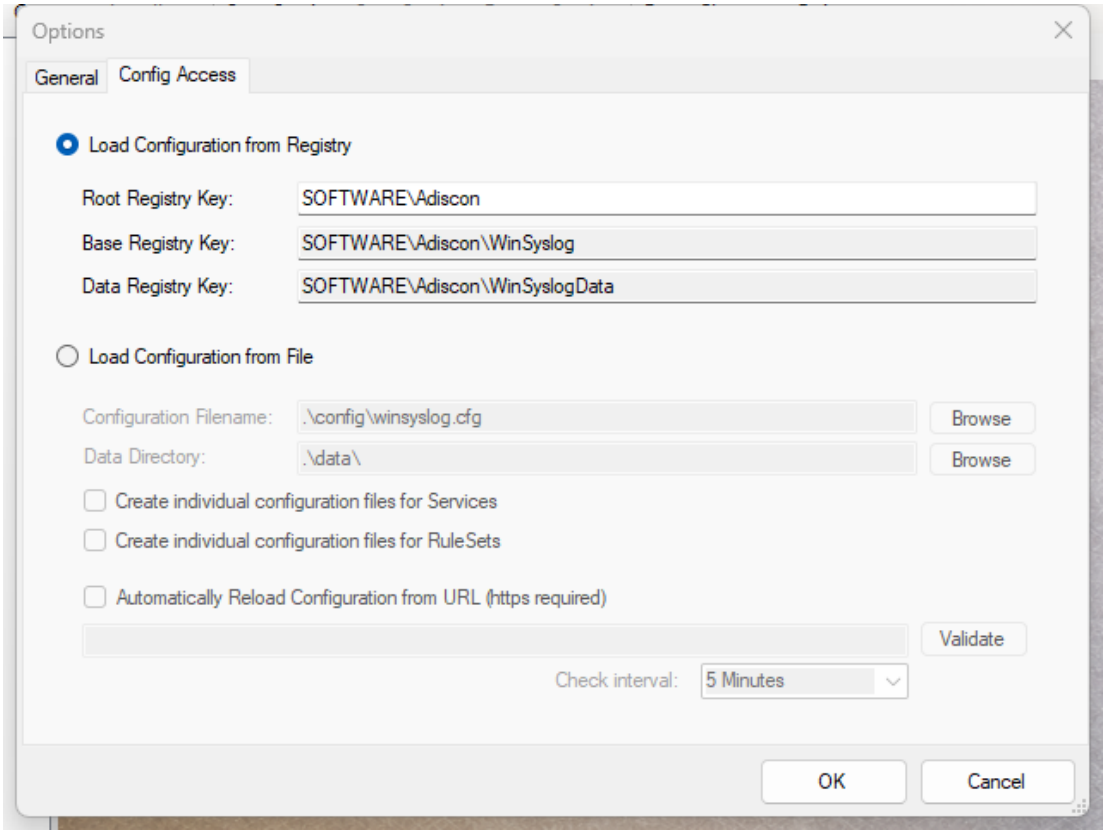
When you change the language, a popup will ask you to reload the configuration client to properly apply the changes and load with the set language.

**Load SNMP Mibs Description (Slows OID Browser down first time)**

If enabled, load SNMP Descriptions from MIB files (Client starts a little slower on startup).

**Automatically sort Datagrids**

Datagrids are used in certain areas within the configuration objects. You can change the default sorting behavior from ascending to descending here.




*WinSyslog Config Access tab*

**Load Configuration from Registry**

The Configuration Client can be switched to a different registry path for configuration. The registry path change can be made permanent here. The changed registry path is the saved within the Parameters key of the Service.

**Load Configuration from File**

Alternatively, you can configure the service to load the configuration from a file. You can set the paths with the two fields below.

When enabled, the configuration will always be backed up before applying the new configuration. The backup consists of the last iteration and will be placed in the same directory.

**Create individual configuration files for Services**

Can only be enabled when “Load Configuration from File” is enabled. When enabled, the Services section of the configuration will be put into a separate file.

**Create individual configuration files for RuleSets**

Can only be enabled when “Load Configuration from File” is enabled. When enabled, the RuleSet section of the configuration will be put into a separate file.

**Automatically Reload Configuration from URL (https required)**

Only possible if File Configuration Mode is used.

If enabled, the configuration will be reloaded from a remote https location. Please note that a valid SSL certificate is required, or if custom certificates are used they have to be imported on the local machine properly.

If the remote configuration file can be downloaded from the configured location and differs from the current configuration, it will be installed automatically and the service will reload itself.

**Check interval**

Specifies how often the service will check for remote configuration files. Please keep in mind that the configuration needs to be downloaded each time from the remote https url for comparison with the local one. We do not recommend to use a value lower then 5 minutes.

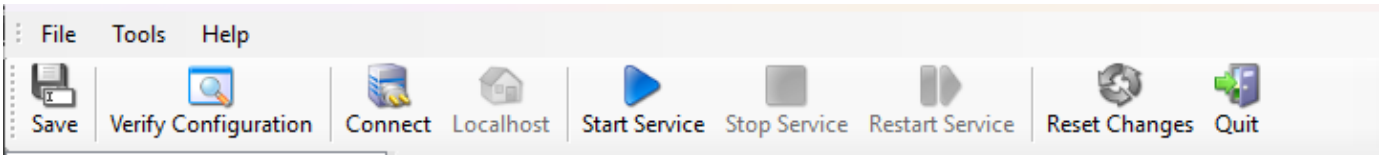
## Client Tools

There are tools within the configuration client that you can use to test certain services or debug the application in general. Some can be found in the Tools menu.

### Service control

From the **main window** toolbar: **Start**, **Stop**, **Restart**, **Kill**, and **Open Windows Services**. **Kill** ends an orphan process when the service manager state is inconsistent. When the service is paused during configuration reload, stop and restart may be unavailable until reload completes.

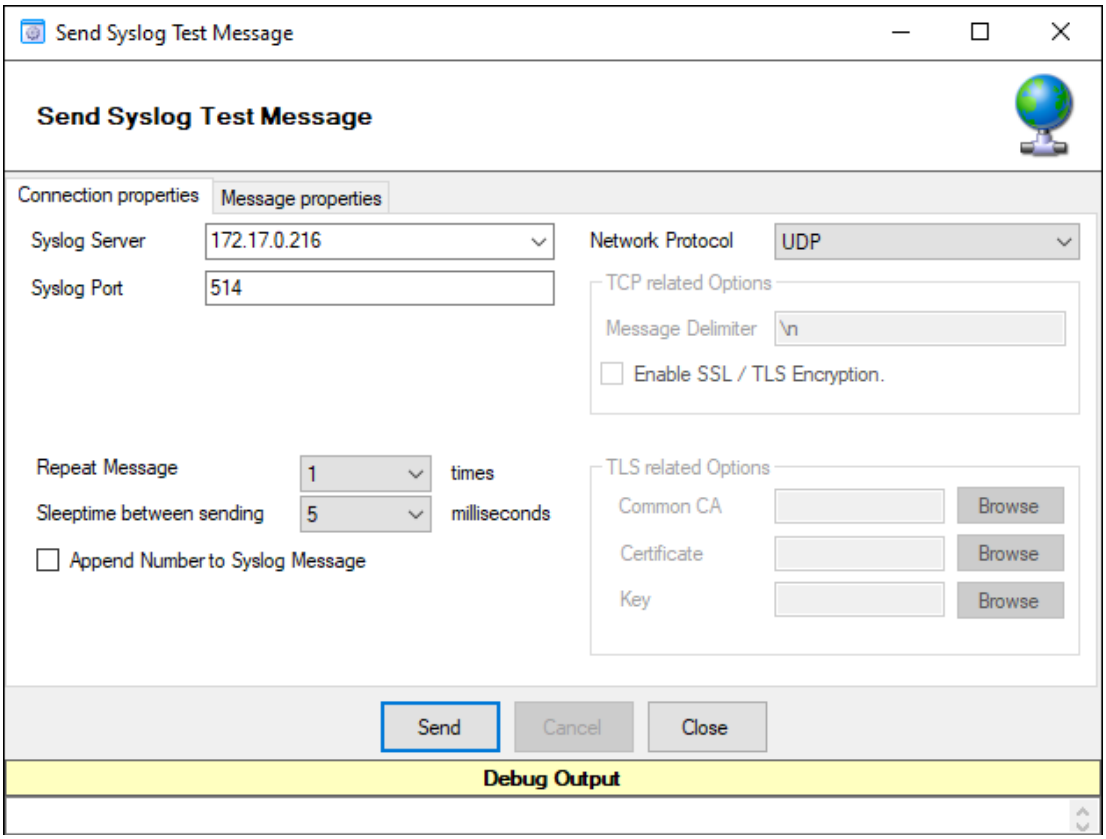
The property window toolbar (shown below) provides **Start**, **Stop**, **Restart**, and **Reset Changes** while editing a service or action. **Kill** and **Open Windows Services** are on the main window toolbar only.




See User Interface changes for **Reset to default** (selected item vs entire configuration), YAML import/export, and the WinSyslog User Interface.

### Syslog Test Message

Opens a new windows which can send syslog test messages to Syslog Servers. This can also be opened within the configuration window of a Syslog service.




- Syslog Test Message Connection properties - UDP*

**Syslog server**

The hostname or ip address of the target Syslog server.

**Syslog Port**

The port that should be used to connect to the target Syslog server.

**Repeat Message**

How often you want to repeat the test message. Can be configured from 1 to 1000.

**Sleeptime between sending**

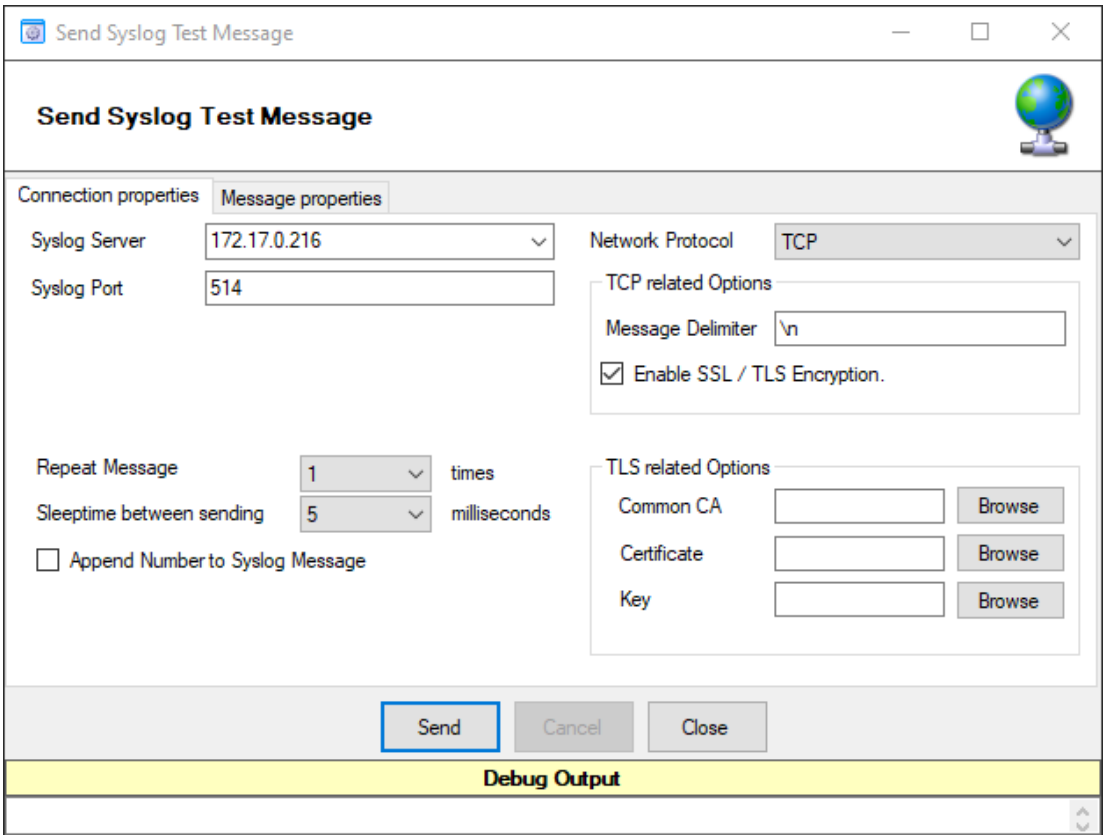
When using TCP, you can use 0ms. For UDP we recommend 1-5ms as sleeptime between sending syslog messages. Otherwise package loss can happen.

**Append Number to Syslog Message**

If sending multiple messages, enable this option in order to add a syslog number at the end of the message.

**Network Protocol**

Which network protocol should be used, either UDP or TCP can be selected.




- Syslog Test Message Connection properties - TCP*

**Message Delimiter (TCP related Options)**

When using TCP protocol, a message delimiter (separator) can be configured which is a simple linefeed by default.

**Enable SSL/TLS Encryption (TCP related Options)**

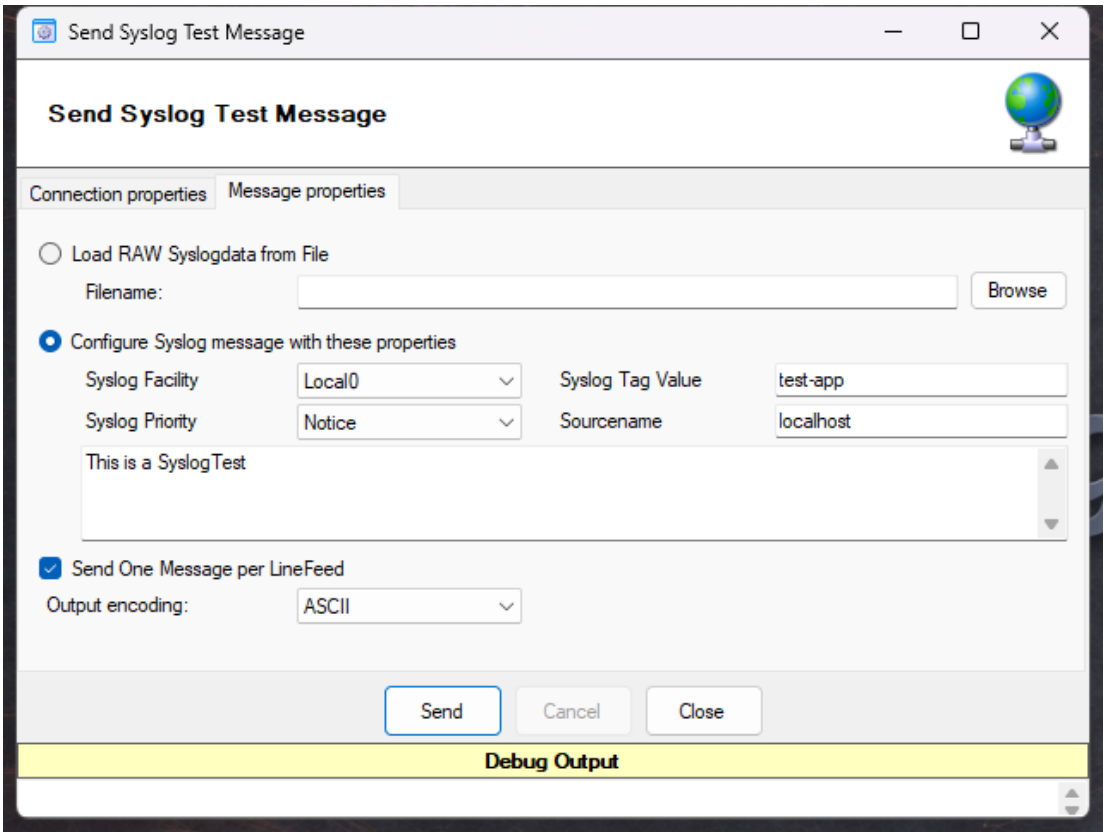
Check this option to enable the TLS related Options.

**TLS related Options (TCP related Options)**

Select common CA: Select the certificate from the common Certificate Authority (CA), the syslog receiver should use the same CA.

Select Certificate: Select the client certificate (PEM Format).

Select Key: Select the keyfile for the client certificate (PEM Format).




*WinSyslog Syslog Test Message properties*

**Load RAW Syslogdata from File**

You can choose to load raw syslogdata from file using this option. When loading UTF8 data make sure to set the Output encoding format from ASCII to UTF8. And if your file contains multiple syslog messages make sure that - Send One Message per LineFeed - is checked.

**Configure Syslog message with these properties**

Choose this if you want to configure all properties of the syslog message manually.

**Send one Message per LineFeed**

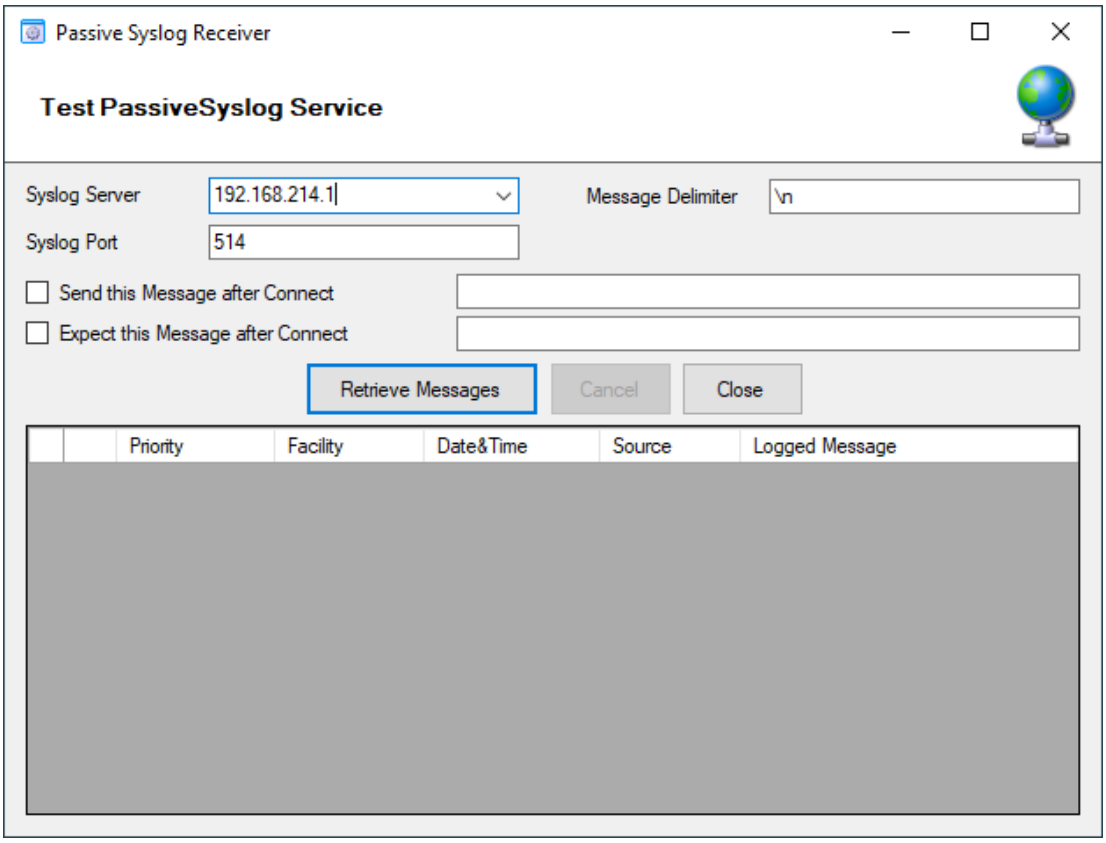
Check if your syslogdata contains multiple syslog messages divided by line feeds

**Output encoding**

Select the Output encoding you wish to use. When using UTF8, the UTF8 BOM is automatically prepended.

### Passive Syslog Receiver

Opens a new windows to test Passive Syslog Servers. This can also be opened within the configuration window of a Passive Syslog service.




- Test Passive Syslog Service*

**Syslog server**

The hostname or ip address of the target passive Syslog server.

**Syslog Port**

The port that should be used to connect to the target passive Syslog server.

**Message Delimiter**

The message delimiter (separator) used to split syslog messages which is a simple linefeed by default.

**Send this Message after Connect**

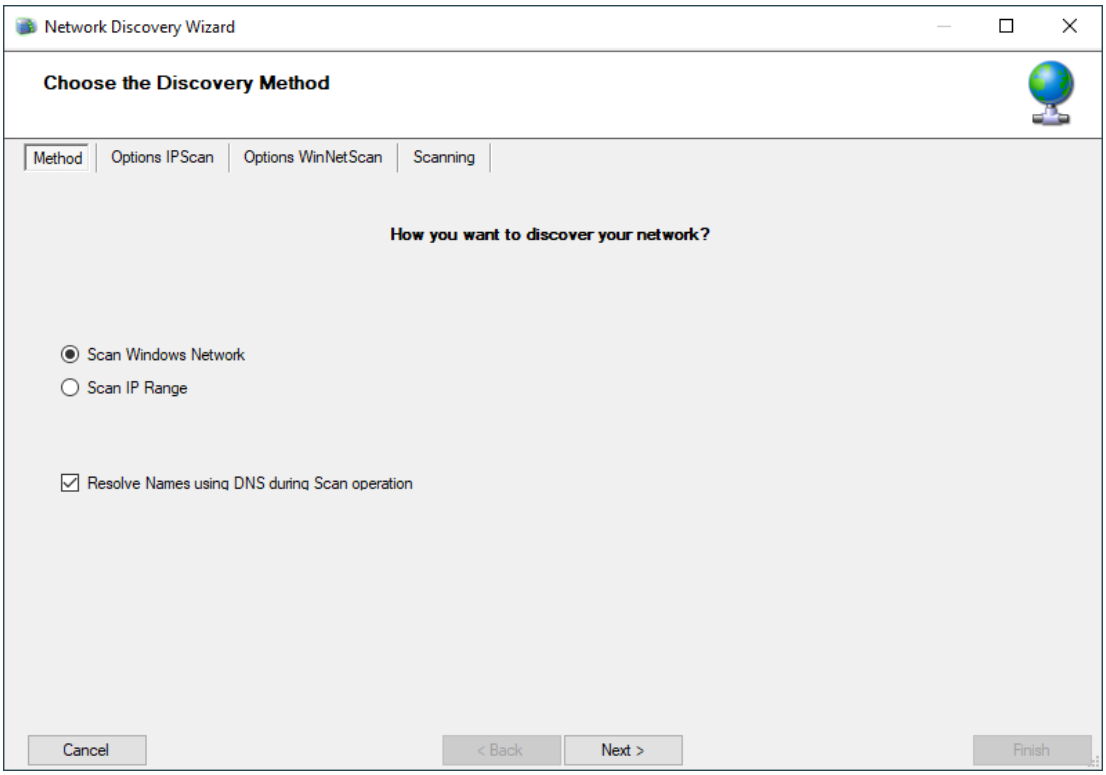
If required, configure a custom message that is send to the server after connect.

**Expect this Message after Connect**

If required, configure a custom message that is expected by the sender when the server response to our custom message.

### Network Discovery

Opens up a Wizard that will help you discover devices in your local network. Once the wizard has scanned your network, it will show Windows compatible devices it has found. Please note that this will require Windows Management Instrumentation (WMI) access to the remote machines which may be disabled in Windows Firewalls by default.



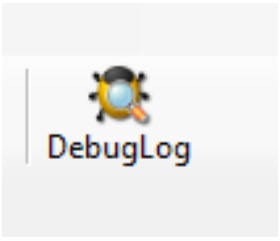

- Network Discovery - Choose the discovery Method*

### Kill Service

When stopping a service, and it does not shutdown in the time period, you can use this function to forcefully stop the service. The service process will be killed if possible.

### DebugLog

The **DebugLog** Button will be available if Debug Logging is enabled in your Debug Options




- DebugLog*

When clicked, a new Logviewer window will be opened. The Debug Logviewer can load, parse, and analyze debug log-files from the service.




- Logviewer (Debug)*

**Debugfile**

Will automatically be set to your configured debug file. You can also choose other saved debug-files for analysis.

**Load**

When Load is clicked, the Logviewer will load lines as configured in the initial debug-lines field. When loading all log-lines on a large debug log-file, this may take a while. While the Load button is grayed out, the Logviewer will continue to read data from the debug log as it is being written.

**Stop**

Stop continuous loading of the debug log.

**Reset**

Will reset all loaded log-lines from memory and clear the debug data-grid.

**Init Debuglines**

The amount of log-lines you want to read the first time.

**Show debug messages for**

Once the debug-log is processed, the Logviewer will automatically add filters for objects like services, rulesets, rules, and actions. You can use this select box to filter by them.

**Filtering (bottom bar)**

At the bottom of the Logviewer window, you can filter the debug-log for Thread (ID), Priority, internal Facility, and Functions. You can also filter for words or word sequences. The view will automatically be refreshed once you changed a filter.

## Using File based configuration

**Working with File based Configurations**

Support for running the Service from file based configuration may be interesting for environments where you want to minimize registry access to a minimum or you want to manually edit the configuration without using the configuration client every time.

The Adiscon Configuration format is quiet simple. In the following description, all the configuration options will be explained in detail.

**Adiscon Configuration format explained**

Our configuration format is something between JSON and XML but hold at a very simple level.

**Variables**

All variables start with a dollar ($). Name and Value of a variable are separated by the FIRST space character. Everything else behind the first space will be considered as the Value. A line feed terminates the value. If your configuration value contains line feeds, you have to replace them with “\n” or “\r\n”. A single backslash can be used to escape brackets ( { and } ).

**Comments**

All lines starting with a sharp (#) at the beginning will be ignored.

**File Includes**

Sample

```
includeconfig my-subconfigfiles-*.cfg*
```

The includeconfig statement will include either a single file or many files based on a filename pattern. In this sample all Files starting with “my-subconfigfiles-” and ending with “.cfg” will be included into the configuration. It is possible to create your own custom file structure with includes. The configuration client will be able to load and show your custom file structure, however it will not be able to maintain (save) it. We support a maximum include depth of up to 10 levels when using the includeconfig statement.

**General Options**

Sample

```
general(name="[name]") {
 $nOption 1
 ...
}
```

All options between the brackets will be loaded as variables into the general configuration object. The name attribute field specifies the general configuration block name. The brackets start and end an object block.

**Services**

All possible configuration parameters are named within the detailed services documentation.

Sample Service configuration:

```
input(type="[ID]" name="[name]") {
 $var1 Value1
 $var2 Value2
 ...
}
```

The brackets start and end a service block. All variables between the brackets will be loaded into the service configuration. The name attribute specifies the service display name. The type attribute contains the service type ID. It can be one of the following types:

```
1       = Syslog
2       = Heartbeat
3       = EventLog Monitor V1 (Win 2000 / XP / 2003 )
4       = SNMP Trap Listener
5       = File Monitor
8       = Ping Probe
9       = Port Probe
10      = NTService Monitor
11      = Diskspace Monitor
12      = Database Monitor
```

```
13      = Serialport Monitor
14      = CPU Monitor
16      = MonitorWare Echo Request
17      = SMTP Probe
18      = FTP Probe
19      = POP3 Probe
20      = IMAP Probe
21      = IMAP Probe
22      = NNTP Probe
23      = EventLog Monitor V2 (Win VISTA/7/2008 or higher)
24      = SMTP Listener
25      = SNMP Monitor
26      = RELP Listener
27      = Passive Syslog Listener
1999998 = MonitorWare Echo Reply
1999999 = SETP Listener
```

**RuleSets**

All possible configuration parameters are named within the detailed actions documentation.

Sample

```
ruleset(name="[name]" expanded="[on/off]") {
 rule(name="[name]" expanded="[on/off]" actionexpanded="[on/off]"
 ThreatNotFoundFilters="[on/off]" GlobalCondProperty="[on/off]"
 GlobalCondPropertyString="" ProcessRuleMode="[0/1/2]"
ProcessRuleDate="[uxtimestamp") {
       action(type="[ID]" name="[name]") {
               $var1 Value1
               $var2 Value2
               ...
       }
       filter(nTabSelection="0") {
               $nOperationType AND
               $PropertyType NOTNEEDED
               $PropertyValueType NOTNEEDED
               $CompareOperation EQUAL
               $nOptionalValue 0
               $nSaveIntoProperty 0
               $szSaveIntoPropertyName FilterMatch
       }
    }
}
```

The brackets start and end a ruleset block. The attributes of a Ruleset are self-explainable. Within a RuleSet, you can have Rules. The attributes of Rules are also self-explainable and partially Global Conditions that are equal to the options found in the Filter dialog. Within a Rule you can one Basefilter. This Basefilter again can have child filters it and these child filters can have child filters again. All “expanded” settings are optional and only important for the client treeview.

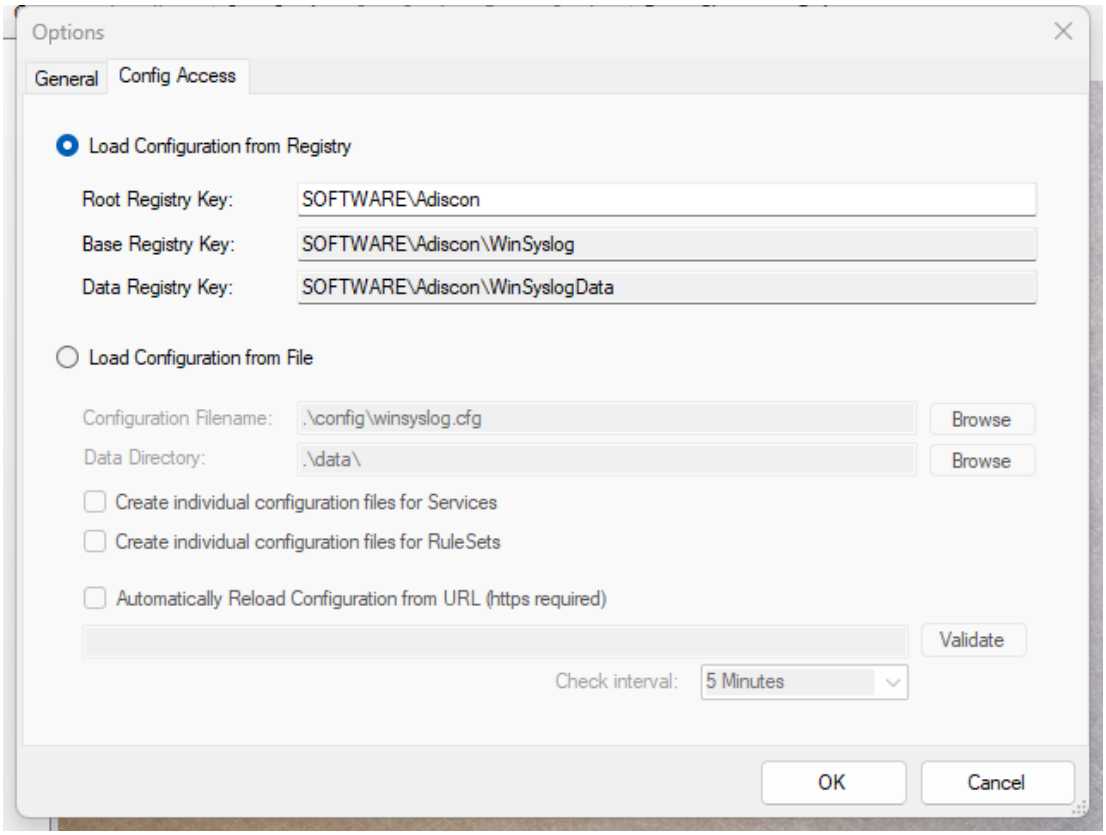
Within a Rule you can have Actions. The brackets start and end an action block. All variables in an action block between the brackets will be loaded into the action configuration. The name attribute specifies the service display name. The type attribute contains the action type ID. It can be one of the following types:

```
1000 = ODBC Database
1001 = Send Syslog
1008 = Net Send (legacy; removed from 26.07 configuration client)
1009 = Start Program
1011 = Send SETP
1012 = Set Property
1013 = Set Status
1014 = Call RuleSet
1015 = Post Process
1016 = Play Sound
1017 = Send to Communication Port
```

```
1021 = Send SNMP
1022 = Control NT Service
1023 = Compute Status Variable
1024 = HTTP Request
1025 = OleDB Database
1026 = Resolve Hostname
1027 = Send RELP
1028 = Send MS Queue
1029 = Normalize Event
1030 = Syslog Queue
1032 = HTTP REST Output
1033 = Message Ringbuffer
```

**How to enable file based configuration?**

To switch from registry to file configuration mode, all you need to do is to go the “Config Access” tab in the Configuration “Client Options” and switch from “Load Configuration from Registry” to “Load Configuration from File” mode. Once you accept the change, the Client will ask you if you want to export the current loaded configuration into the file. Hit YES if you want to do so and NO if already have an existing configuration file. The configuration client will reload itself automatically after this.




*WinSyslog file-based configuration options*

**Create individual configuration files for Services**

When enabled, the configuration client will create separated configuration files for each configured service. The main configuration file will then use the includeconfig statement to include all these configuration files by using a pattern. When deleting a service, its configuration file will be deleted as well.

**Create individual configuration files for RuleSets**

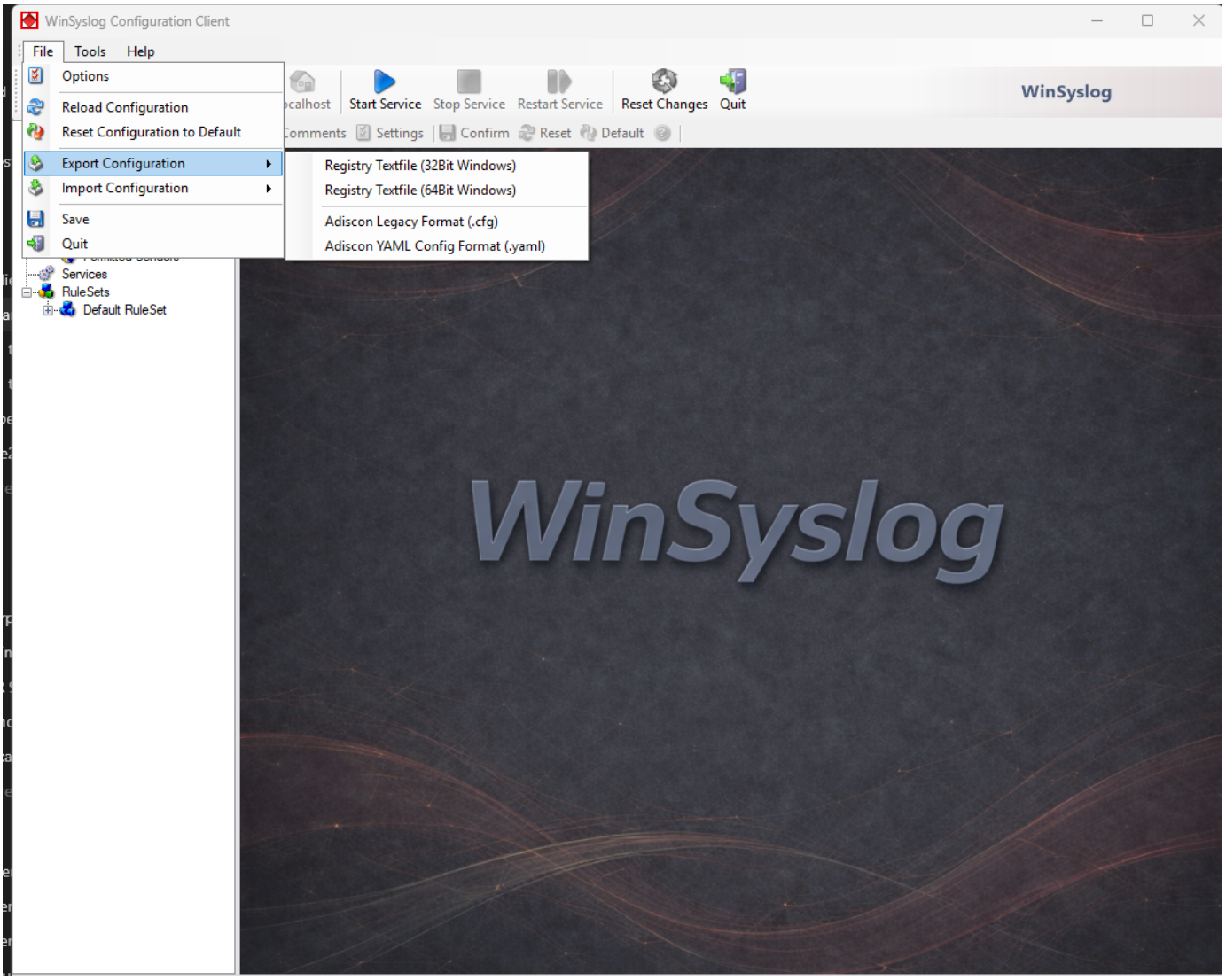
When enabled, the configuration client will create separated configuration files for each configured ruleset. The main configuration file will then use the includeconfig statement to include all these configuration files by using a pattern. When deleting a ruleset, its configuration file will be deleted as well

## Adiscon YAML configuration format

The configuration client can import and export **Adiscon YAML Config Format** (`.yaml`) files in addition to the legacy `.cfg` format (**Adiscon Legacy Format** in the File menu).

New file-based deployments should use YAML configuration. The legacy `.cfg` Adiscon Config Format remains supported for compatibility with existing installations, test fixtures, support workflows, and external tooling, but it is not required for new Adiscon product deployments.

**Import / export:** Use **File** → import or export and select the YAML format. Select legacy CFG only when your workflow requires that format.




*WinSyslog File menu with YAML import and export commands*

**Unsupported blocks:** If you open a configuration that contains blocks this client version does not understand (for example newer service features), the client skips them and may show a warning. Saving from an older client can drop those sections. See What happens when the client opens a configuration with unsupported blocks?.

**License file path in general options:**

```
general(name="License") {
 $szLicenseV2Path C:\\ProgramData\\Adiscon\\MonitorWare\\license.alic
}
```

**Send RELP window size example:**

```
action(type="1027" name="Forward RELP") {
 $nSelpWindowSize 128
 ...
}
```

**HTTP REST Output example:**

```
action(type="1032" name="REST Out") {
 $szProfile
 $szServer api.example.com
 $szPath /v1/events
 $bUseTLS true
 $szMethod POST
 ...
}
```

## General Options

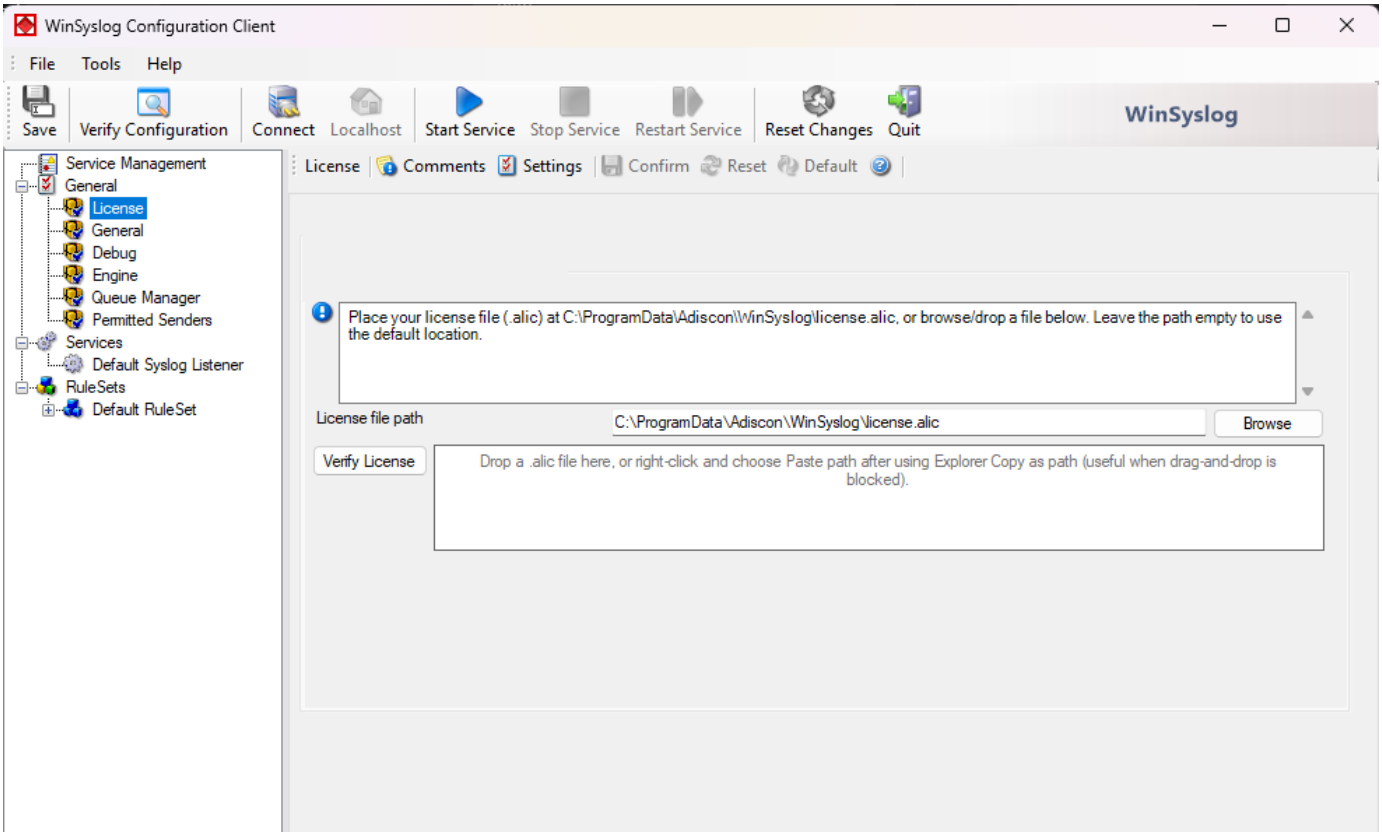
In this chapter, you find the general option settings.

## License

Licensing options are on **General** -> **License** in the configuration client. Current builds use a `license.alic` license file.

### License File

Use this page to apply `license.alic`.




**File Configuration field:**

szLicenseV2Path

**Description:**

Optional path to the license file. When empty, the service uses the default location:

`%ProgramData%\Adiscon\WinSyslog\license.alic`

You can browse for the file, drag-and-drop it onto the client, or paste a file path. The license is validated and applied when you save the configuration.

Current configuration clients use the license file path only. They do not read or save legacy registration-name or numeric license-key configuration values. Update an existing configuration to use `szLicenseV2Path` and a `license.alic` file.

The main window status bar shows license status (for example, the licensed organization). A version mismatch between the license file and the installed service may show a warning banner until you apply a matching license.

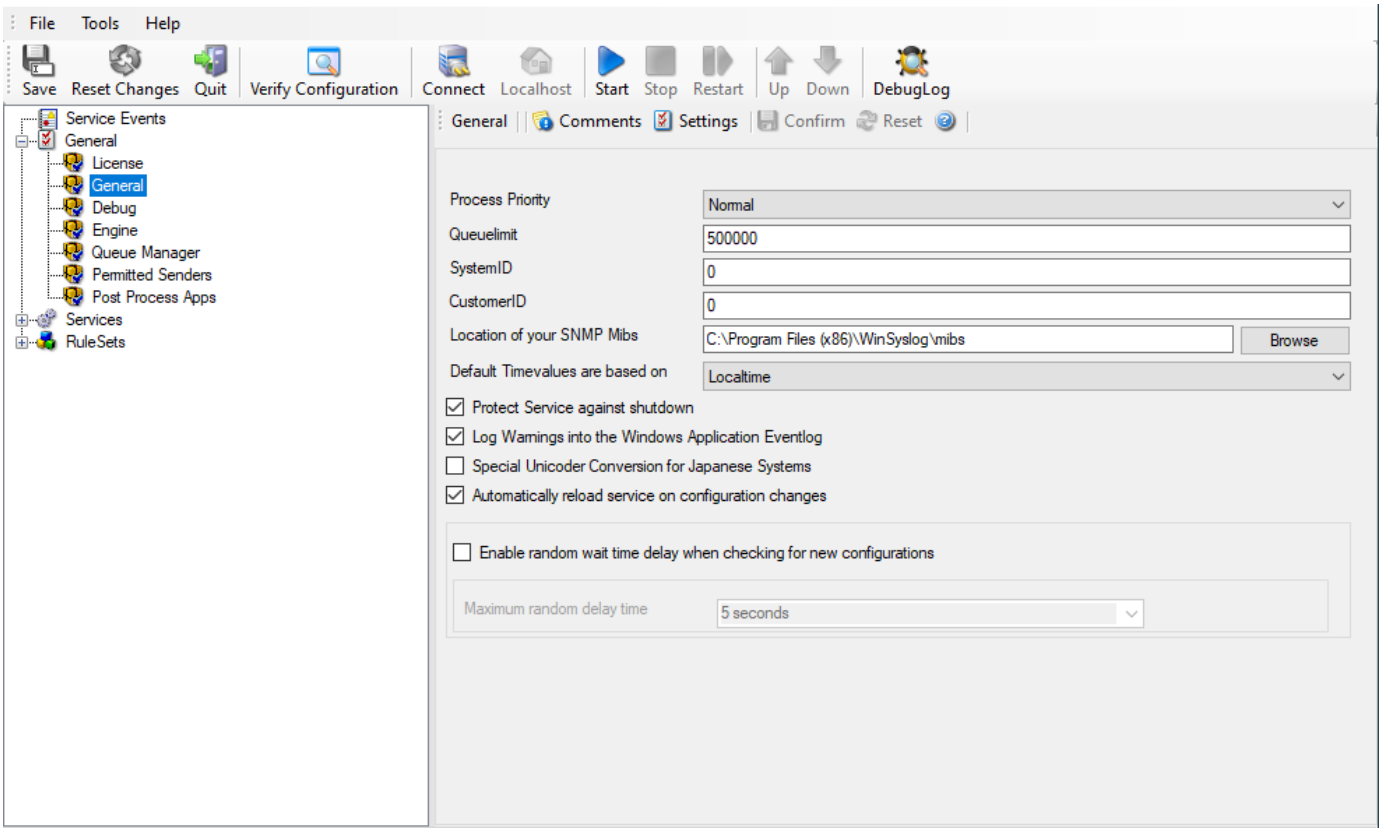
### Verify License

Use this control to verify that the configured license is valid before or after you save.

If a newly issued file reports that it is not active yet, wait until the activation time shown by the client and then verify it again.

## General

The General Options available on this form are explained below:




### Process Priority

**File Configuration field:**

nProcessPriority

**Description**

Configurable Process Priority to fine-tune application behavior.

### Queue Limit

**File Configuration field:**

nQueueLimit

**Description**

The applications keeps an in-memory buffer where events received but not yet processed are stored. This allows the product to handle large message bursts. During such burst, the event is received and placed in the in-memory queue. The processing of the queue (via rulesets) itself is de-coupled from the process of receiving. During traffic bursts, the queue size increases, causing additional memory to be allocated. At the end of the burst, the queue size decreases and the memory is freed again.

Using the queue limit, you can limit that maximum number of events that can be in the queue at any given time. Once the limit is reached, no further enqueuing is possible. In this case, an old event must first be processed. In such situations, incoming events might be lost (depending on the rate they come in at). A high value for the queue size limit (e.g. 500,000) is recommended, because of the risk of message loss. It is also possible to place no limit on the queue. Use the value zero (0) for this case. In this case, the queue size is only limited by virtual memory available. However, we do not recommend this configuration as it might cause the product to use up all available system memory, which in turn could lead to a system failure.

### SystemID

**File Configuration field:**

nSystemID

**Description**

SystemID is of type integer to be used by our customer. In addition, it is user configurable.

### CustomerID

**File Configuration field:**

nCustomerID

**Description**

CustomerID is of type integer provided for customer ease. For example if someone monitors his customer's server, he can put in different CustomerIDs into each of the clients. Let us say someone monitors servers A and B. A has 5 servers all of them with CustomerID = 1 and B has 2 servers all of them with CustomerID = 2. Both A and B happen to have a server named "SERVER". Together with the customerID, these machines are now uniquely identifiable. This is user configurable.

### Location of your SNMP MIBs

**File Configuration field:**

szMIBSPath

**Description**

Click the Browse button to search for your MIBs location or enter the path manually. The Client and Service will read all files from this directory automatically on startup.

### Default Timevalues are based on

**File Configuration field:**

nTimeMode

**Description**

The general options of each product (EventReporter, WinSyslog and WinSyslog) contain a setting for the "Default Timevalues are based on". This setting can be set to Localtime which is default and UTC (Universal Coordinated Time). This setting has an effect on:

- Send Email Action: The date in the email header is affected
- Start Program Action: Time parameters in the command line are affected
- Write File Action: Time properties in the file name are affected
- Filter Engine: If you filter by weekday or time fields, localtime does affect the filter result

For information about how to get local-time output, see the WinSyslog FAQ entry on Default Timevalues.

### Protect Service against shutdown

**File Configuration field:**

nProtectAgainstShutdown

**Description**

When enabled (default), the Agent will not stop processing the internal queue when it is stopped. **Please note that it will remain in the stopping state then.**

### Shutdown protection timeout

This capability is available starting with **26.08**.

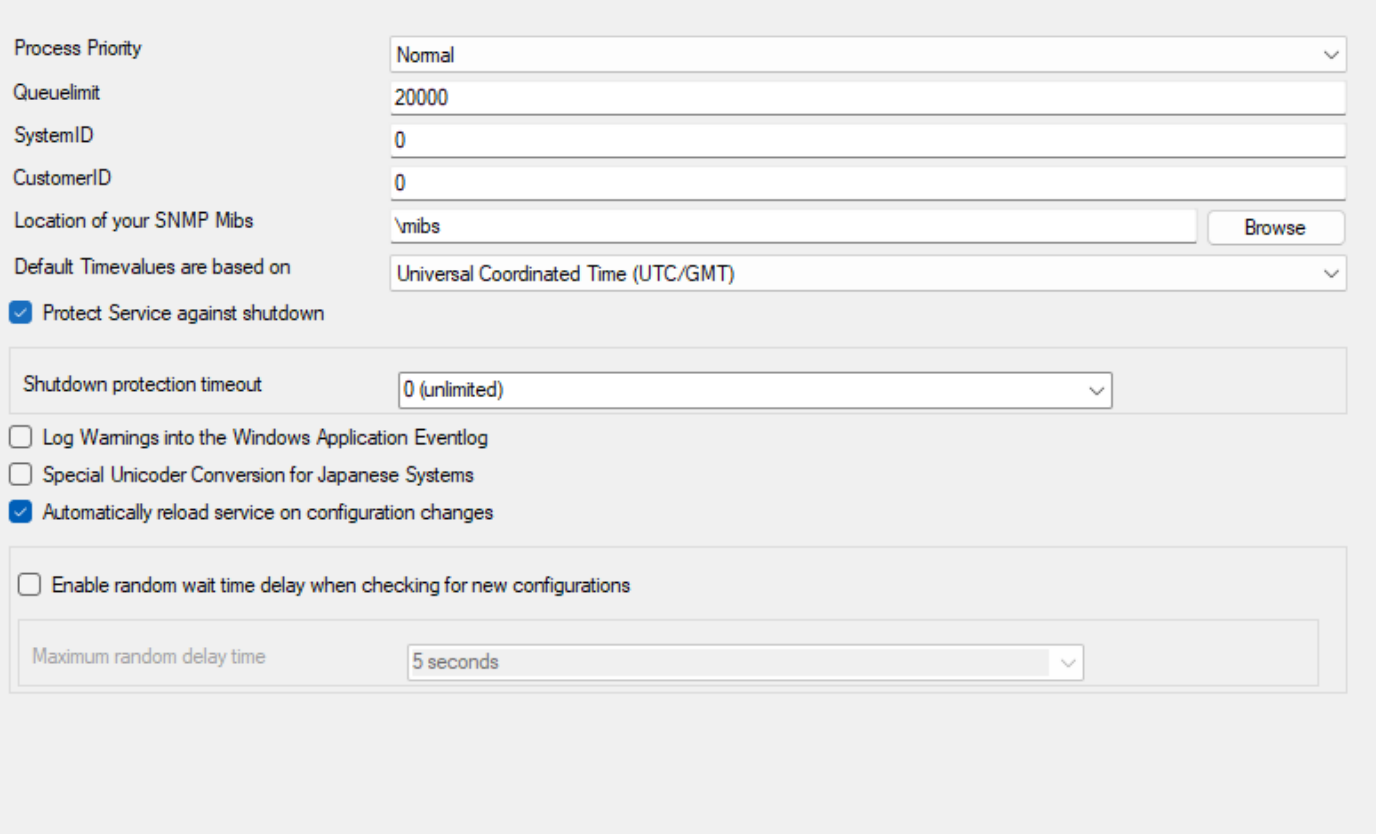
**File Configuration field:**

nShutdownProtectionTimeout

**Description**

This setting applies only when **Protect Service against shutdown** is enabled. Enter the maximum time in milliseconds for the complete protected shutdown. Set it to `0` to wait without a deadline for cooperative queue processing to finish.

A positive timeout bounds shutdown time, but messages that are still in memory or have not been stored durably can be lost when that deadline is reached.




*Shutdown protection enabled with the timeout set to 0 (unlimited).*

### Log Warnings into the Windows Application Eventlog

**File Configuration field:**

nEnableEventlogWarnings

**Description**

The Service will also log Warnings into the Windows Application Eventlog, and so be more verbose for troubleshooting. Default is enabled.

### Special Unicoder Conversion for Japanese Systems

**File Configuration field:**

nJapanStringHandling

**Description**

This is a historical option for older multibyte systems from the time when UTF8 was not known yet. If enabled, whenever text is being converted from 16 Bit wide character to 8 Bit character, the conversion is done with bit masking in order to avoid broken encoding. **For today modern systems, we do NOT recommend to enable this option.**

### Automatically reload service on configuration changes

**File Configuration field:**

nEnableAutoConfigReload

**Description**

When enabled (default), the service will detect configuration changes and reload it’s core automatically. This feature only works if the latest Client Application is used for configuration. It will also work if you are using the file based configuration method and update the configuration file. It will not work if you are using the service in console mode unless you send any input to the console.

### Enable random wait time delay when checking for new configurations

**File Configuration field:**

bAutoReloadRandomDelay

**Description**

When enabled, a random delay (with the configured maximum) will be added between new configuration checks.

### Maximum random delay time

**File Configuration field:**

nAutoReloadDelayTime

**Description**

The maximum for this random delay is 24 hours. The random delay has no affect on the service control anymore.

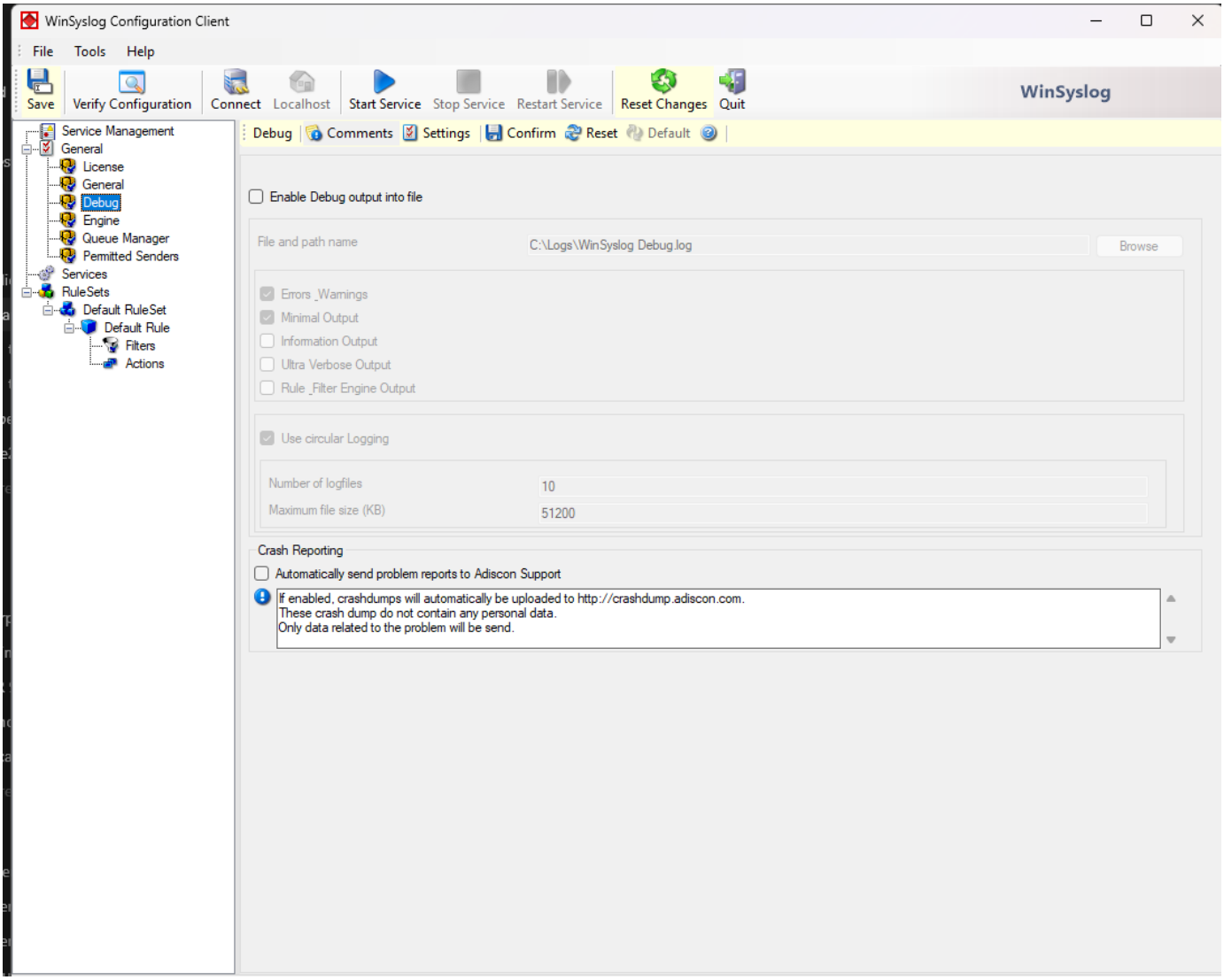
## Debug

This tab can be used to debug rule bases. Especially with complex bases, it might be necessary to learn what application is internally doing while it is processing them. With the debug log, the service tells you some of these internal workings.

Other than rule basis testing, the debug log is also helpful when contacting Adiscon support. An Adiscon support engineer might ask you to set the debug log to a specific level while doing troubleshooting.

### Note

Debug logging requires considerable system resources. The higher the log level, the more resources are needed. However, even the lowest level considerable slows down the service. As such, we highly recommend turning debug logging off for normal operations.




*WinSyslog Debug options*

### Enable Debug output into file

**File Configuration field:**

nEnableDebugOutput

**Description**

If checked, the debug log is enabled and written as the service operates. If unchecked, no debug log is written. For performance reasons, it is highly recommended that this box is unchecked during normal operations.

### File and path name

**File Configuration field:**

szDebugFileName

**Description**

The full name of the log files to be written. Please be sure to specify a full path name including the drive letter. If just the file and/or path name is specified, that information is local to the service default directory. As this depends on a number of parameters, it might be hard to find the actual log file. So for consistency purposes, be sure to specify a fully qualified file name including the drive.

Note: If the configured directories are missing, they are automatically created by application i.e. the folder specified in “File and Path Name”.

### Debug Levels

**File Configuration field:**

nDebugErrors, nDebugMini, nDebugInternal, nDebugUltra, nDebugRuleEngine

**Description**

These checkboxes control the amount of debug information being written. We highly recommend only selecting “Errors & Warnings” as well as “Minimum Debug Output” unless otherwise instructed by Adiscon support.

### Use circular Logging

**File Configuration field:**

nCircularLogging

**Description**

Support for circular debug logging has been added as the debuglog can increase and increase over time. This will avoid an accidental overload of the hard disk. Of course you can also customize the amount of files used and their size or disable this feature.

### Automatically send problem reports to Adiscon Support

**File Configuration field:**

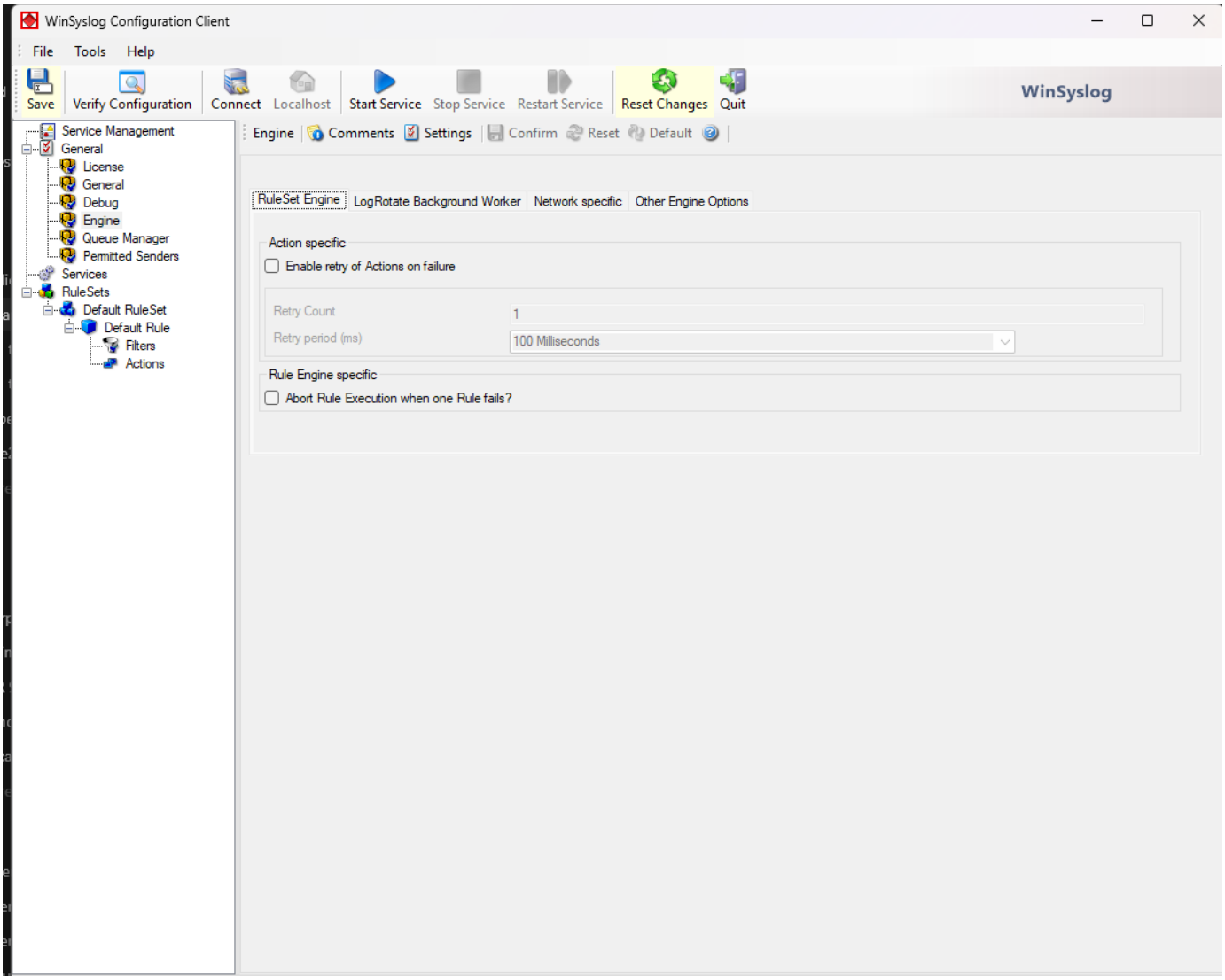
nReportCrash

**Description**

If enabled, problem reports will automatically be uploaded to `http://crashdump.adiscon.com`. A problem report is generated if the service internally stops working for some unknown reason. The reports are small dumpfiles which do not contain any personal data and will help us find and fix the problem. Also the dumpfiles are very small and do not exceed 256 Kbyte. In most cases only 32Kbyte data is send.

## Engine

The Engine specific Options are explained below:




*WinSyslog RuleSet Engine tab*

### Action specific

#### Enable retry of Actions on failure

**File Configuration field:**

nEnableRetry

**Description**

If enabled, the Agent retries Actions on failure (until the retry counter is reached). Note that the Event error 114 will only be written if the last retry failed, previous error's will only be logged in the debug log (with the error facility). Note that you can customize the Retry Count and the Retry Period in ms as well.

### Rule Engine specific

#### Abort Rule Execution when one Rule fails?

**File Configuration field:**

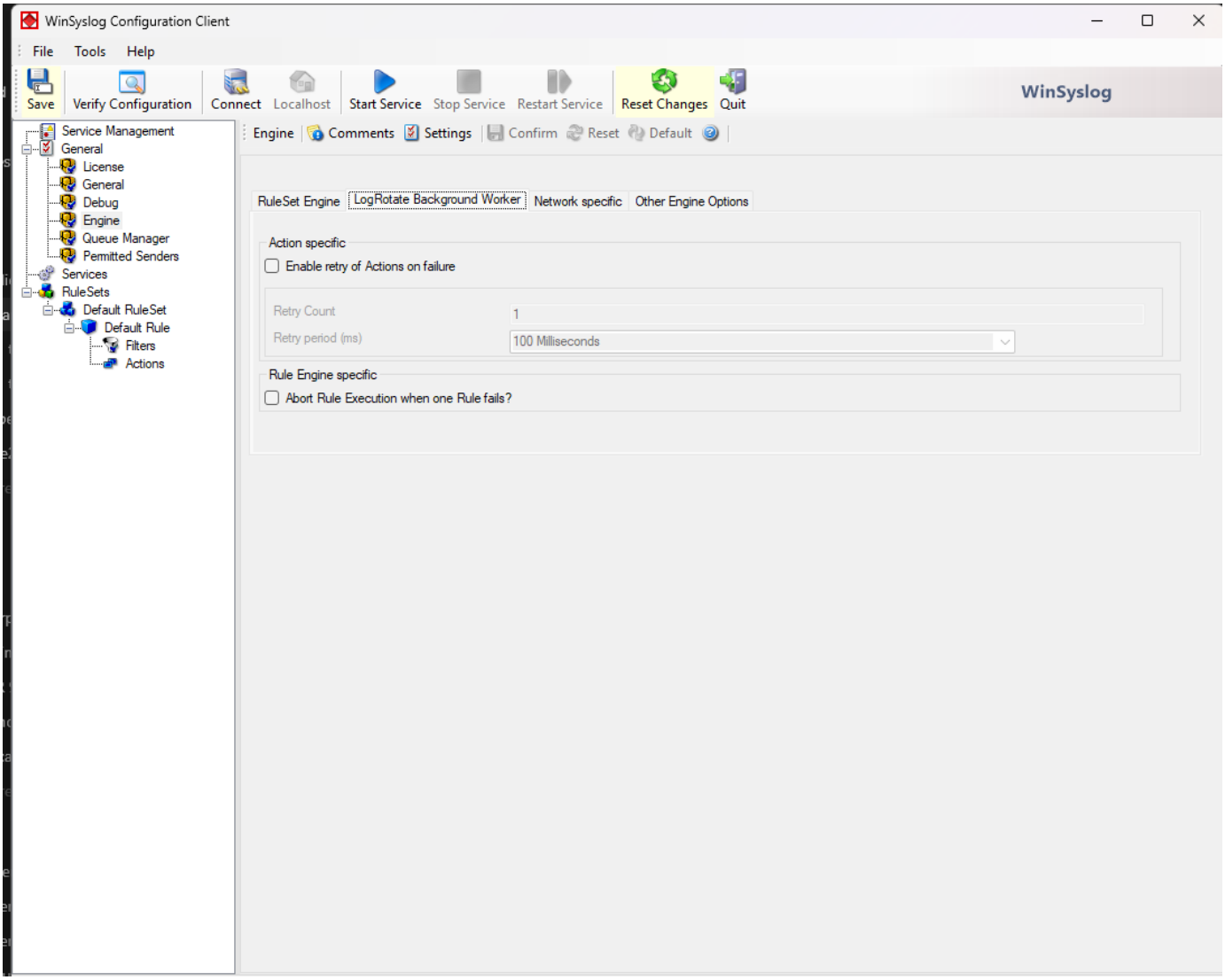
bAbortRuleOnFailure

**Description**

If checked, and an action fails, the execution will be aborted. If unchecked, and an action fails, simply the next action in this rule will be executed.

### LogRotate Background Worker

The LogRotate Background worker checks all configured File Actions periodically and handles the time based log rotation settings, if enabled.




*WinSyslog LogRotate Background Worker tab*

### Wait time between Logrotation checks

**File Configuration field:**

nLogRotateWorkerSleepTime

**Description**

Defines how often the logrotate background worker thread checks all configured actions to see if any logfiles need to be rotated based on time related rotate conditions.

### Wait time if log rotation is running on service shutdown

**File Configuration field:**

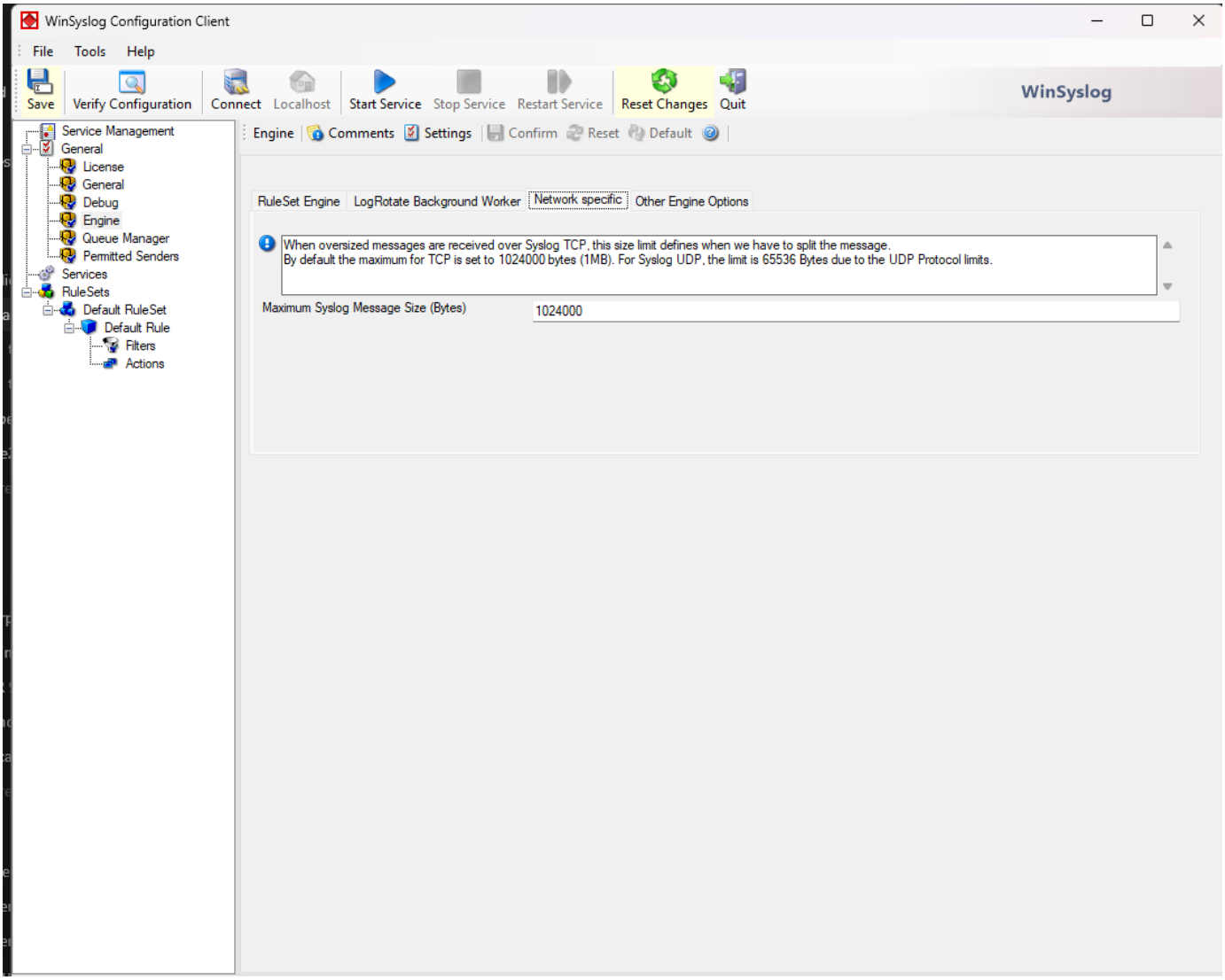
nLogRotateWorkerStopWaitTimeout

**Description**

When service is being shutdown, this defines how much time the logrotate background worker thread has left to finish its log rotations before the service exits.

If a queued or already active detached rotation does not finish within this bounded wait time, the unfinished detached work is saved and resumed on the next startup instead of extending shutdown indefinitely. For the full guarantee, see **`../shared/faq/log-rotation-guarantees`**.

### Network specific Options




*WinSyslog Network specific Options tab*

### Maximum Syslog Message Size (Bytes)

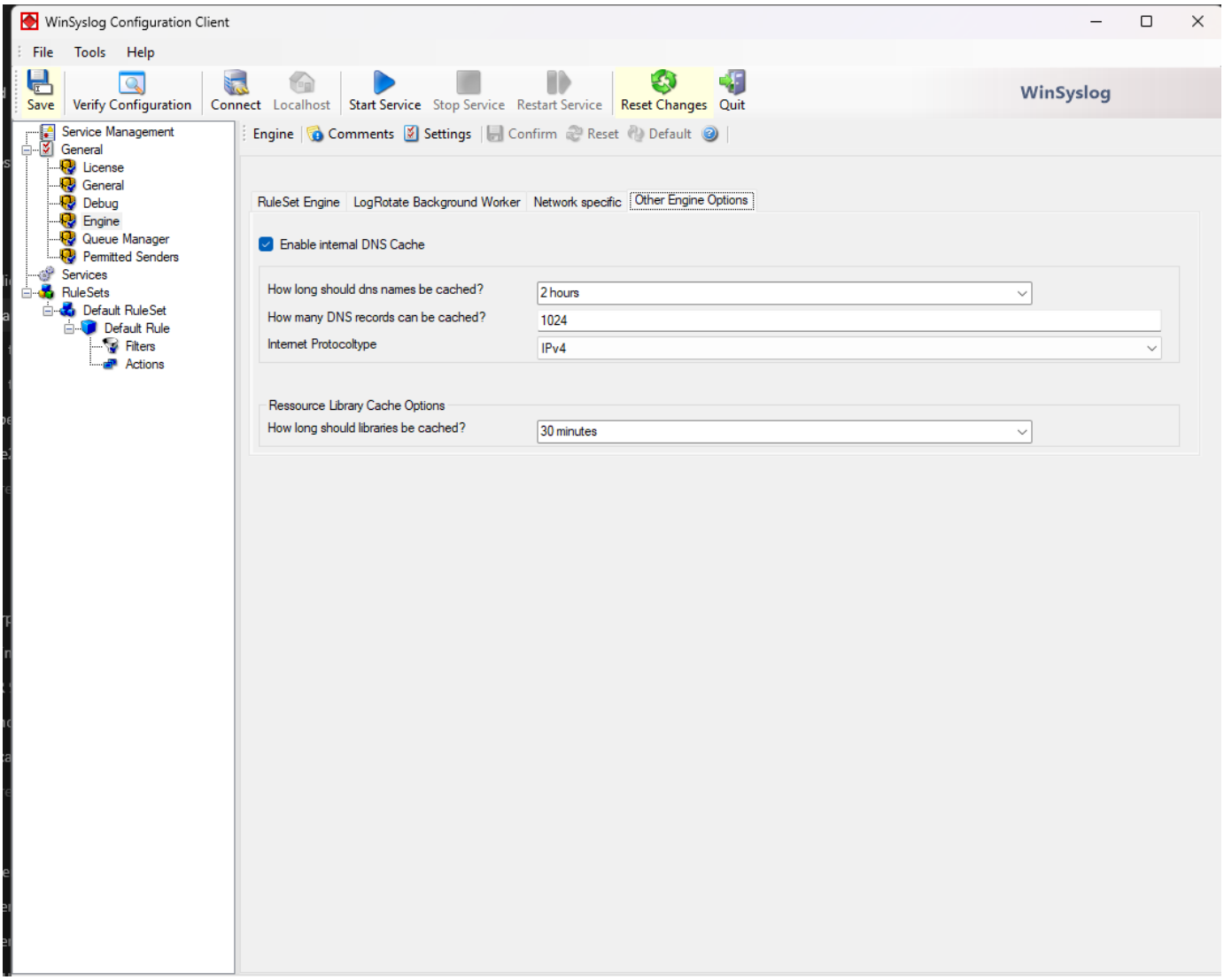
**File Configuration field:**

nSyslogMaxMessageSize

**Description:**

Configurable message size limit for Syslog TCP messages. The default is 1MB which is far more as defined in Syslog RFC's. If a syslog message exceeds the size limit, it will be split into multiple messages.

## Other Engine Options




*WinSyslog Other Engine Options tab*

### Enable internal DNS Cache

**File Configuration field:**

nEnableDNSCache

**Description**

The DNS cache is used for reverse DNS lookups. A reverse lookup is used to translate an IP address into a computer name. This can be done via the resolve hostname action. For each lookup, DNS needs to be queried. This operation is somewhat costly (in terms of performance). Thus, lookup results are cached. Whenever a lookup needs to be performed, the system first checks if the result is already in the local cache. Only if not, the actual DNS query is performed and the result then stored to the cache. This greatly speeds up reverse host name lookups.

However, computer names and IP addresses can change. If they do, the owner updates DNS to reflect the change. If we would cache entries forever, the new name would never be known (because the entry would be in the cache and thus no DNS lookup would be done). To reduce this problem, cache records expire. Once expired, the record is considered to be non-existing in the cache and thus a new lookup is done.

Also, cache records take up system memory. If you have a very large number of senders who you need to resolve, more memory than you would like could be allocated to the cache. To solve this issue, a limit on the maximum number of cache records can be set. If that limit is hit, no new cache record is allocated. Instead, the least recently used record is overwritten with the newly requested one.

### How long should DNS names be cached?

**File Configuration field:**

nDNSCacheTime

**Description**

This specifies the expiration time for cache records. Do not set it too high, as that could cause problems with changing names. A too low-limit results in more frequent DNS lookups. As a rule of thumb, the more static your IP-to-hostname configuration is, the higher the expiration timeout can be. We suggest, though, not to use a timeout of more than 24 to 48 hours.

### How many DNS records can be cached?

**File Configuration field:**

nDNSCacheLimit

**Description**

This is the maximum number of DNS records that can be cached. The system allocates only as many memory, as there are records required. So if you have a high limit but only few sending host names to resolve, the cache will remain small. However, if you have a very large number of host names to resolve, it might be useful to place an upper limit on the cache size. But this comes at the cost of more frequent DNS queries. You can calculate about 1 to 2 KBytes per cache record.

### Internet Protocoltype

**File Configuration field:**

nDNSInetProtocol

**Description**

Select if you wish to prefer IPv4 or IPv6 addresses for name resolution. Note that this only has an effect on names which return both, IPv4 and IPv6 addresses.

## Resource Library Cache Options

### How long should libraries be cached?

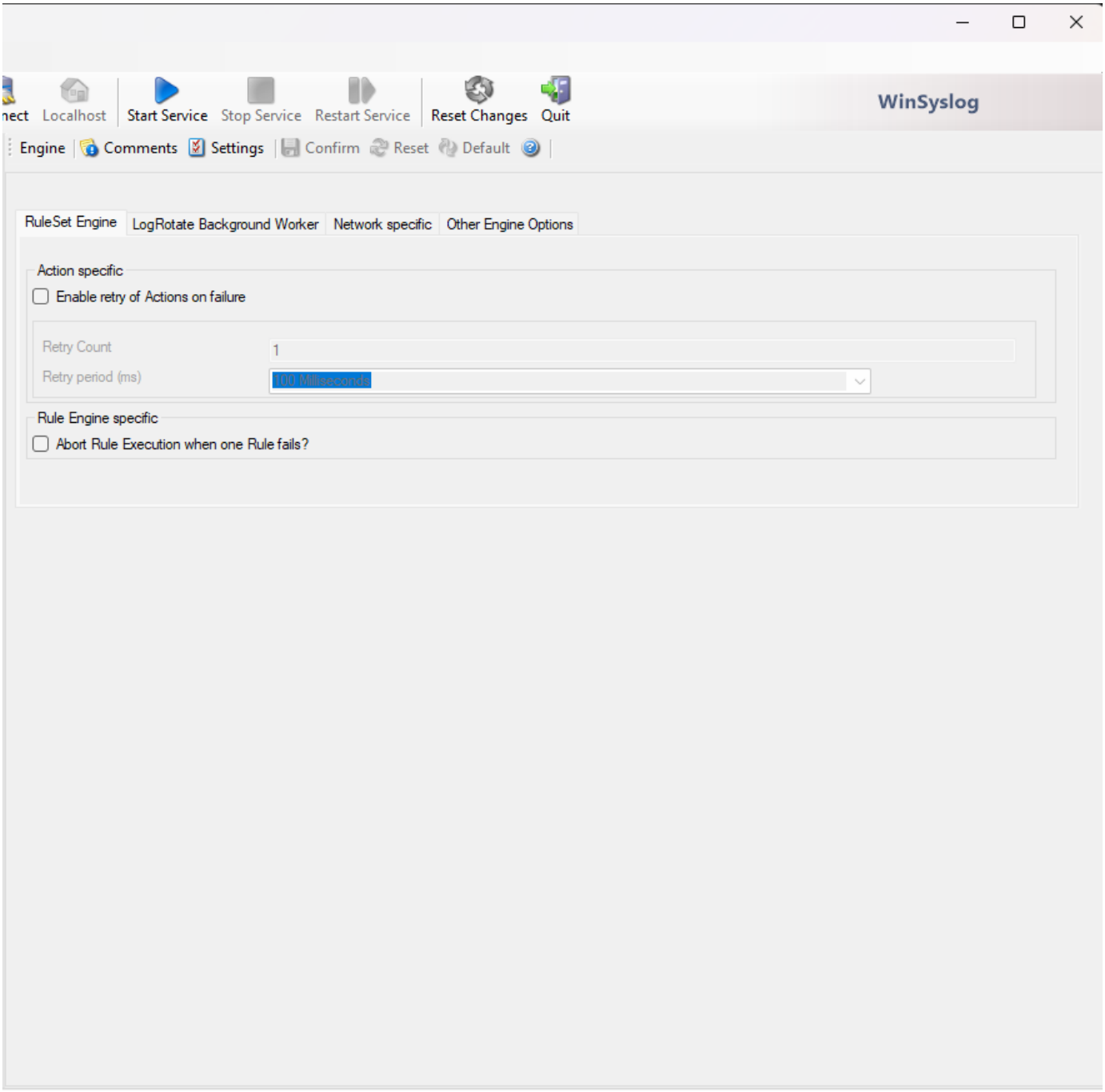
**File Configuration field:**

nLibCacheTimeOut

**Description**

This feature will be mainly useful for EventLog Monitor. For events with the same recurring event sources, this will be a great performance enhancement. The cache will also work for remote system libraries (requires administrative default shares). All libraries will be cached for 30 minutes by default.

## Queue Manager




*WinSyslog Queue Manager options*

### Enable Queue Manager DiskCache

This feature enables the Agent to cache items in its internal queue on disk using a fixed data file.

> **Warning**
>
> Only use this feature if you really need to!

Depending on the speed of your hard disks, it will slow down processing of the actions, in worst case if the machine cannot handle the IO load, the Queue will become full sooner or later. The DiskCache is an additional feature for customers, who for example want to secure received Syslog messages which have not been processed yet.

The diskcache will not cache infounits from services like EventLog Monitor, as this kind of Service only continues if the actions were successfully. All other information sources like the Syslog server will cache its messages in this file. If the Service or Server crashes for some reason, the queue will be loaded automatically during next startup of the Agent. So messages which were in the queue will not be lost. Only the messages which was currently processed during the crash will be lost.

### Enable Queue Manager Diskcache

**File Configuration field:**

nEnableRingBuffer

**Description**

Enable the disk based queue manager. Please read the description about the Queue Manager DiskCache first!

### File and Pathname

**File Configuration field:**

szRingBufferFile

**Description**

As everywhere else, you can define here, where the queue file should be stored.

### Queue File Size

**File Configuration field:**

nRingBufferSize

**Description**

With this slider, the queue size can be set from 1 MB to 2048 MB.

### Processing pointer

**File Configuration field:**

nProcessingLow

**Description**

Points to the current processing position within the queue file.

### Saving pointer

**File Configuration field:**

nSavingLow

**Description**

Points to the last processed position within the queue file.

## Queue Manager specific

### Number of worker threads

**File Configuration field:**

nWorkerThreads

**Description**

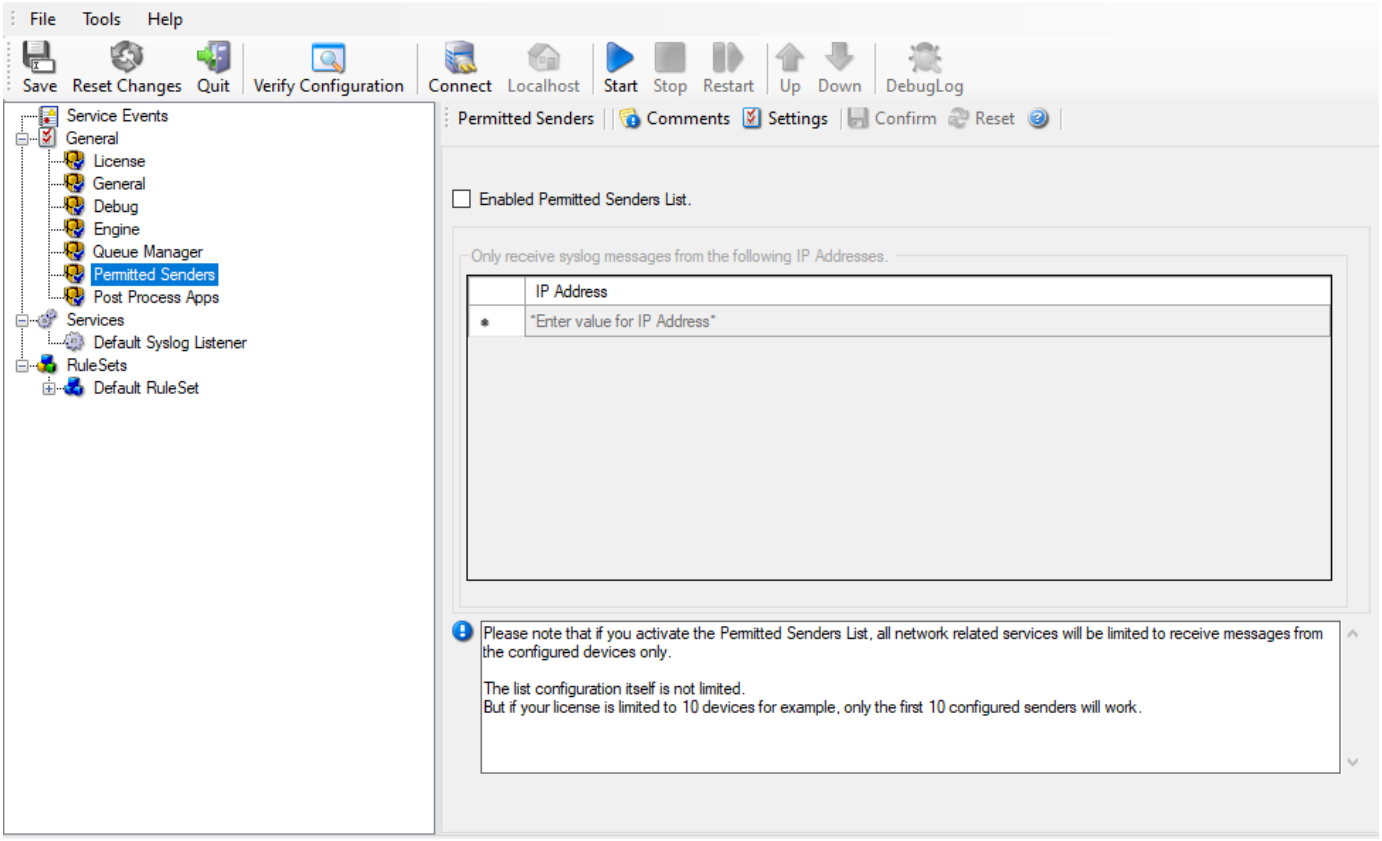
Defines the number of worker background threads that the core engine uses to process its queue.

## Permitted Senders

**Please note that if you activate the Permitted Senders List, all network** related services will be limited to receive messages from the configured devices only.**

**The list configuration itself is not limited.**

**But if your license is limited to 10 devices for example, only the first 10** configured senders will work.**




### Enable permitted Senders List

**File Configuration field:**

nEnablePermittedSenders

**Description**

If this option is enabled, all network related services will be limited to receive messages only from the configured IP Addresses. Please note the list is also limited to your license limit. For example if your license allows 10 devices, only the first 10 configured senders will be allowed.

### Only receive syslog messages from the following IP Addresses

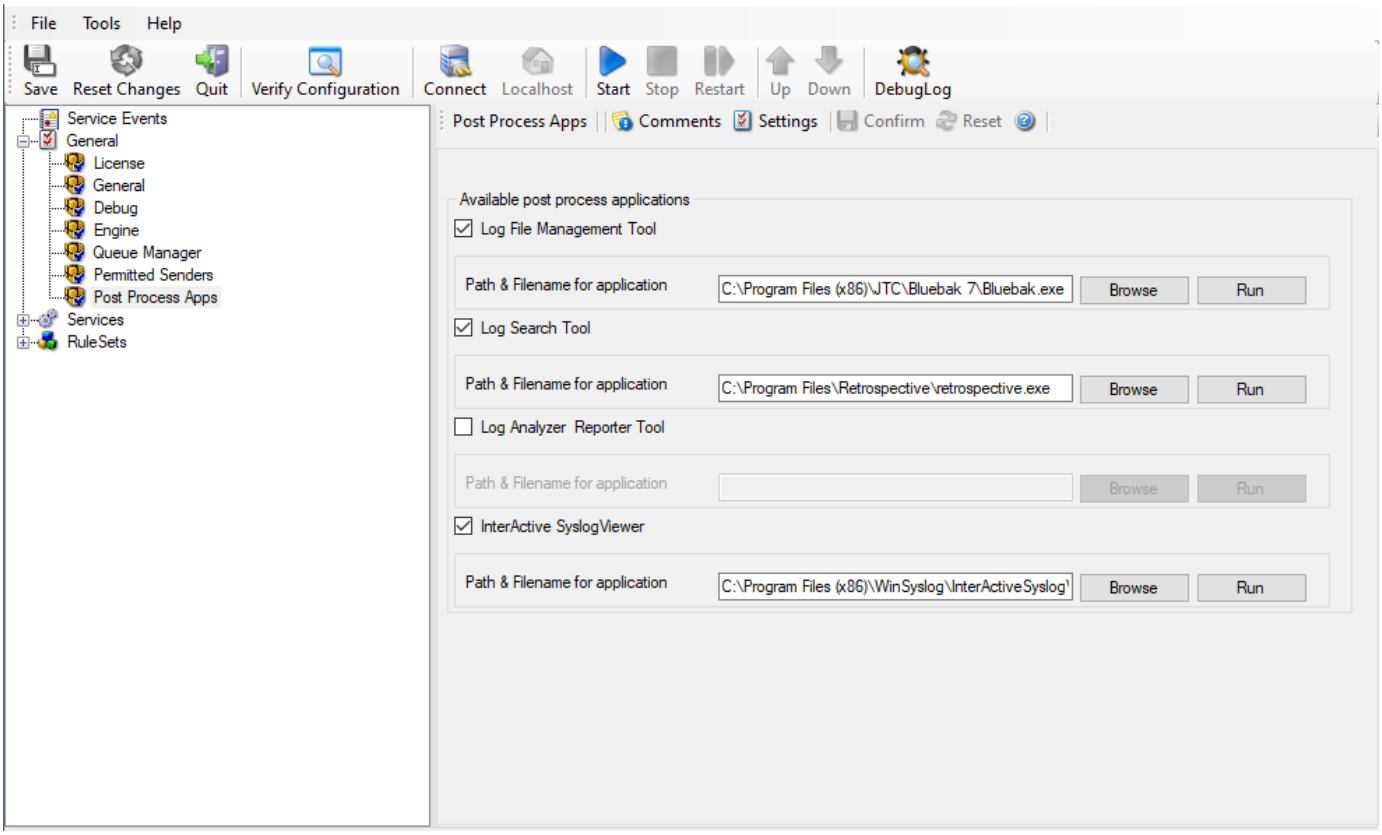
**File Configuration field:**

szIP_[n]

**Description**

This list contains all sender IP Addresses which are allowed to send data to network related services. You can either configure IPv4 or IPv6 Addresses here.

## Post Process Apps




### Available post process applications

#### Log File Management Tool

**File Configuration field:**

nlogfilemgr on/off

**Description**

Enable or disable shortcut for the Log File Management Tool application.

#### Path & Filename for application

**File Configuration field:**

logfilemgr_path

**Description**

Path and Filename of the Log File Management application executable.

#### Log Search Tool

**File Configuration field:**

nlogsearch on/off

**Description**

Enable or disable shortcut for the Log Search Tool application.

#### Path & Filename for application

**File Configuration field:**

logsearch_path C:\Program Files\Retrospective\retrospective.exe

**Description**

Path and Filename of the Log Search application executable.

#### Log Analyzer Reporter Tool

**File Configuration field:**

nloganalyzer on/off

**Description**

Enable or disable shortcut for the LogAnalyzer application.

#### Path & Filename for application

**File Configuration field:**

$loganalyzer_path

**Description**

Path and Filename of the LogAnalyzer application executable.

#### InterActive SyslogViewer

**File Configuration field:**

ninteractive on/off

**Description**

Enable or disable shortcut for the InterActive SyslogViewer application.

#### Path & Filename for application

**File Configuration field:**

interactive_path

**Description**

Path and Filename of the InterActive SyslogViewer application executable.

## Services

Services are the configured WinSyslog inputs and generators. They receive or generate events and pass those events into a ruleset for further processing.

You need at least one active input service. Without a service, WinSyslog does not collect any data.

A few points matter in practice:

- **Actual services** process events.
- **Service defaults** are only templates for creating new service instances.
- Multiple service instances are possible when their ports and settings do not conflict.

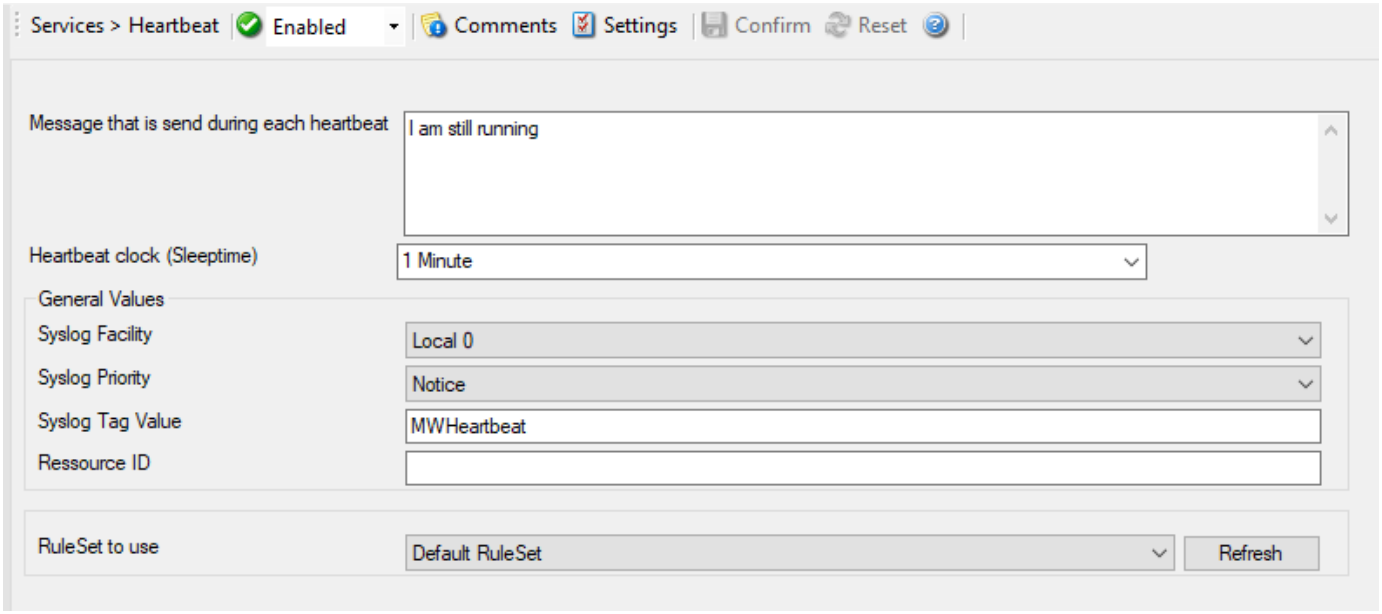
In this manual, **input** is the clearest plain-language concept, while **service** remains the main operational term for the configured WinSyslog object. Some GUI pages use specific labels such as `Syslog server`, `RELP Listener`, and `SETP Server`. Those are exact current service names. Use What do “service”, “input”, “listener”, and “server” mean in WinSyslog? if you need the terminology mapping.

Use the service pages below when you need to choose the right input type or understand what a specific input service receives.

### Basic services

### Heartbeat

The heartbeat process can be used to continuously check if everything is running well. It generates an information unit every specified time interval. That information unit can be forward to a different system. If it does not receive additional packets within the configured interval, it can be assumed that the sender is either in trouble or already stopped running.




- Service - Heartbeat*

#### Message that is send during each heartbeat

**File Configuration field:**

szMessage

**Description:**

This is the message that is used as text inside the information unit. Use whatever value is appropriate. The message text does not have any special meaning, so use whatever value you seem fit.

#### Heartbeat clock (Sleeptime)

**File Configuration field:**

nSleepTime

**Description:**

This is the interval, in milliseconds, that the heartbeat service generates information units in. Please note that the receiving side should be tolerant. The interval specified here is the minimum time between packets. Under heavy load, the interval might be slightly longer. It is good practice to allow twice this interval before the service is considered suspect by the system monitoring the services health.

#### General Values (Common settings for most services)

#### Syslog Facility

**File Configuration field:**

nSyslogFacility

**Description:**

The syslog facility to be assigned to events created by this service. Most useful if the message is to forward to a Syslog server.

### Syslog Priority

**File Configuration field:**

nSyslogPriority

**Description:**

The Syslog priority to be assigned to events created by this service. Most useful if the message is to forward to a Syslog server.

### Syslog Tag Value

**File Configuration field:**

szSyslogTagValue

**Description:**

The Syslog tag value to be assigned to events created by this service. Most useful if the message is to forward to a Syslog server.

### Resource ID

**File Configuration field:**

szResource

**Description:**

The resource id to be assigned to events created by this service. Most useful if the message is to forward to a Syslog server.

### RuleSet to Use

**File Configuration field:**

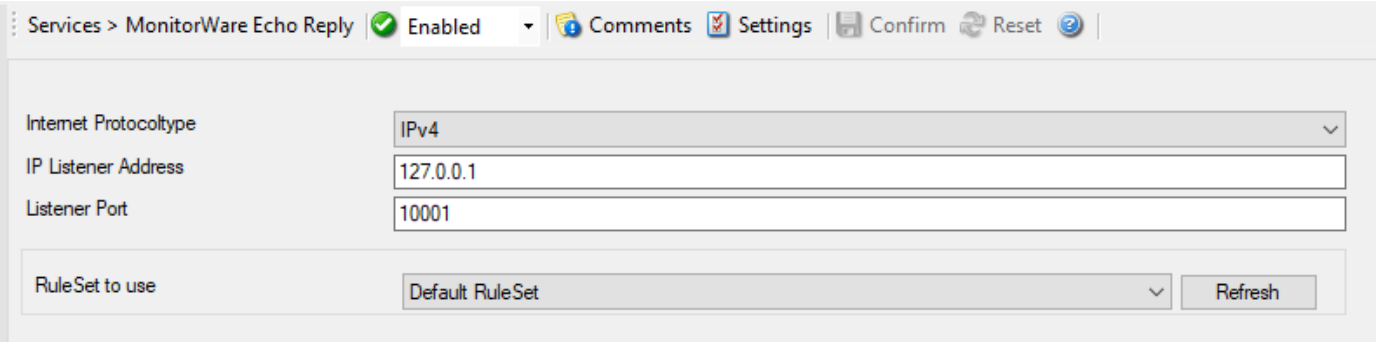
szRuleSetName

**Description:**

Name of the ruleset to be used for this service. The RuleSet name must be a valid RuleSet.

### MonitorWare Echo Reply

The Echo Reply service is used on each of the installed EventReporter/ WinSyslog. A central agent running the WinSyslog is using the echo request and instructs to poll each of the other EventReporter/WinSyslog services. When the request is not carried out successfully, an alert is generated. The MonitorWare echo protocol ensures that always a fresh probe of the remote EventReporter/WinSyslog Service is done.




- Service - MonitorWare Echo Reply*

#### Internet Protocoltype

**File Configuration field:**

nInetType

**Description:**

Select the desired protocol type. IPv4 and IPv6 are available. The IPv6 protocol needs to be properly installed in order to be used. Note that one Service can only handle IPv4 or IPv6, so if you want to use both protocols, you will need to create two separate services.

#### IP Listener Address

**File Configuration field:**

szMyIPAddress

**Description:**

The MonitorWare Echo Reply service can be bound to a specific IP Address. You can either use an IPv4, an IPv6 Address, or a Hostname that resolves to an IPv4 or IPv6 Address. This feature is useful for multihome environments where you want to run different Syslog Servers on different IP Addresses. Please note that the default IP Address `0.0.0.0` means ANY IP Address.

#### Listener Port

**File Configuration field:**

nListenPort

**Description:**

Specify the listener port here.

#### RuleSet to Use

**File Configuration field:**

szRuleSetName

**Description:**

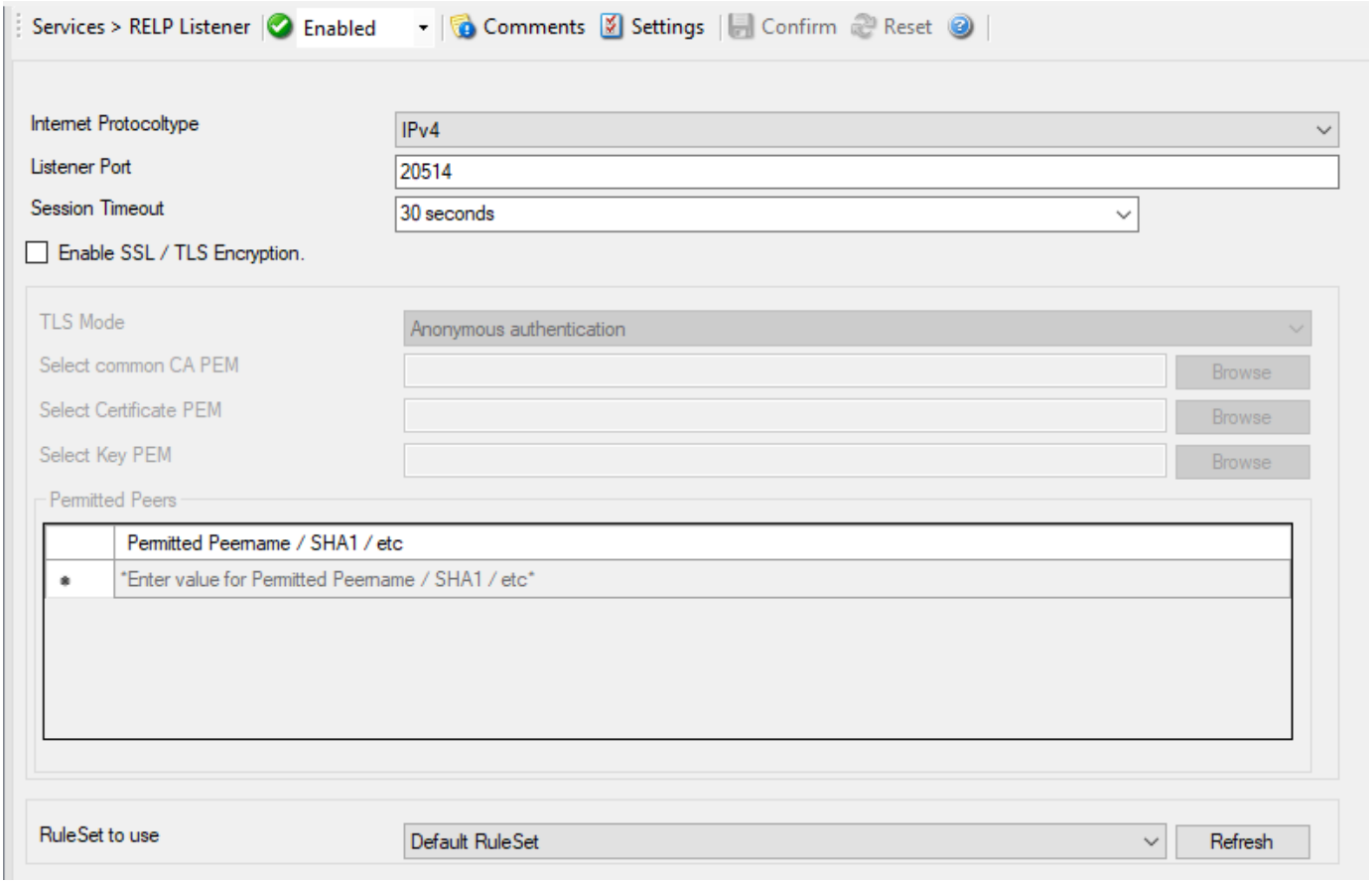
Name of the ruleset to be used for this service. The RuleSet name must be a valid RuleSet.

## Network services

### RELP Listener

The `RELP Listener` service receives messages over the relp protocol. In practice, it is an input service for reliable event delivery and accepts messages from senders that support RELP.

Other than using a different application protocol over TCP, the `RELP Listener` service is functionally equivalent to the `Syslog server` service. It automatically listens on all available IP addresses, including IPv4 and IPv6. This is due to the librelp implementation method.




- Service - RELP Listener*

### Internet Protocoltype

**File Configuration field:**

nInetType

**Description:**

Select the desired protocol type. IPv4 and IPv6 are available. The IPv6 protocol needs to be properly installed in order to be used. Note that one Service can only handle IPv4 or IPv6, so if you want to use both protocols, you will need to create two separate services.

### Listener Port

**File Configuration field:**

nListenPort

**Description:**

The port the `RELP Listener` service listens on. The typical (standard) value is 20514. This should be changed only if there is a definite need for it. Such a need typically arises from security concerns. If the port is changed, all reporting devices must also be configured to use the non-standard port.

### Session Timeout

**File Configuration field:**

nTimeOutSession

**Description:**

It controls how long a session is to be opened from the server side.

### Enable SSL / TLS Encryption

**File Configuration field:**

nUseSSL

**Description:**

This option enables SSL / TLS encryption for the `RELP Listener` service. Please note that with this option enabled, the service only accepts SSL / TLS enabled senders.

### TLS Mode

**File Configuration field:**

nTLSMode

**Description:**

The TLS mode can be set to the following:

**Anonymous authentication** Default option, which means any client certificate will be accepted, or even none.

**x509/name (certificate validation and name authentication)** When this mode is selected, the subject within the client certificate will be checked against the permitted peers list. This means the `RELP Listener` service will only accept the secured connection if it finds the permitted peer in the subject.

**509/fingerprint (certificate fingerprint authentication)** This mode creates a SHA1 Fingerprint from the client certificate it receives, and compares it to fingerprints from the permitted peers list. You can use the debuglog to see fingerprints of client certificates which were not permitted.

### Select common CA PEM

**File Configuration field:**

szTLSCAFile

**Description:**

Select the CA certificate or CA bundle used to validate certificates presented by connecting clients. If you use a CA chain, include the intermediate CA certificates first and the root CA certificate last.

### Select Certificate PEM

**File Configuration field:**

szTLSCertFile

**Description:**

Select the server certificate in PEM format. This is the certificate the `RELP Listener` service presents to connecting clients. If needed, append the intermediate CA certificates after the server certificate so clients can validate the chain.

### Select Key PEM

**File Configuration field:**

szTLSKeyFile

**Description:**

Select the private key in PEM format that matches the server certificate. Passphrase-protected private keys are not supported.

### Permitted Peers

### Permitted Peername / SHA1 / etc

**File Configuration field:**

szIP_[n]

**Description:**

This list contains all permitted peers. If x509/name is used, this can contain parts of the client certificate subject. For example if you have CN = secure.syslog.msg in the certificate subject, you can add “secure.syslog.msg” as permitted peer. When using x509/fingerprint, this list holds a list of permitted SHA1 fingerprints. The fingerprints can either be generated with OpenSSL Tools, or grabbed from the debug logfile. The format is like described in RFC 5425, for example: `SHA1:2C:CA:F9:19:B8:F5:6C:37:BF:30:59:64:D5:9A:8A:B2:79:9D:77:A0`.

### RuleSet to Use

**File Configuration field:**

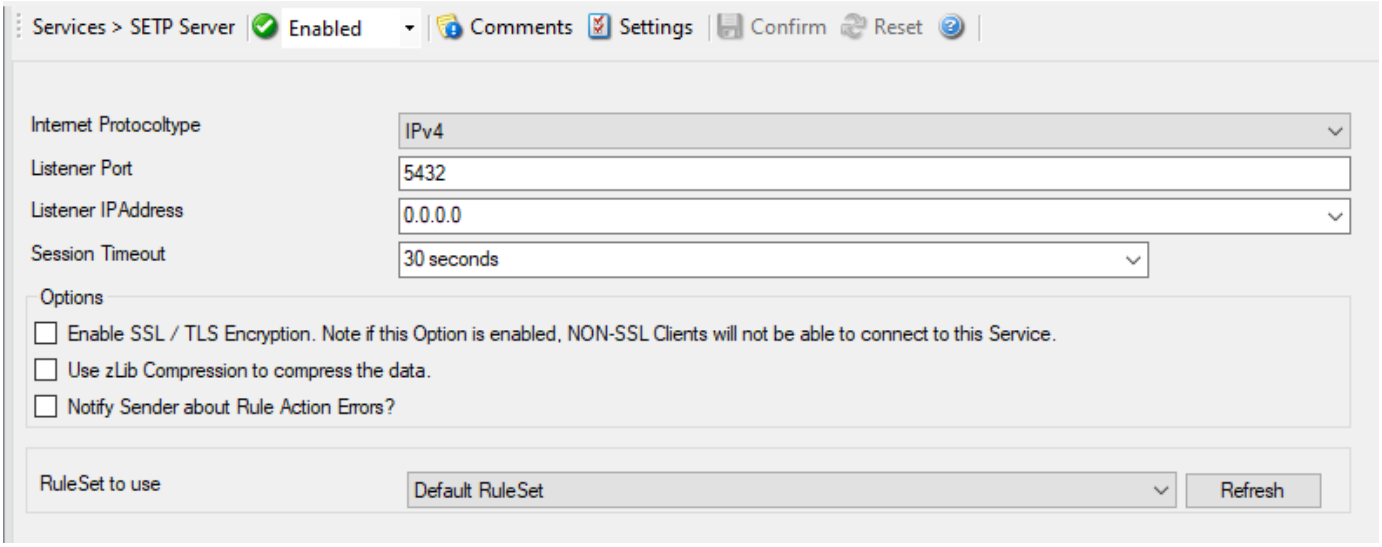
szRuleSetName

**Description:**

Name of the ruleset to be used for this service. The RuleSet name must be a valid RuleSet.

### SETP Server

Configures the `SETP Server` service. In practice, this is an input service that receives setp events from other Adiscon systems. There are only few configuration options, as SETP takes the original message from the sender and uses the exact settings that the sender was configured for. No alteration occurs on the receiving side, so no values need to be configured for the message format.




- Service - SETP Server*

#### Internet Protocoltype

**File Configuration field:**

nInetType

**Description:**

Select the desired protocol type. IPv4 and IPv6 are available. The IPv6 protocol needs to be properly installed in order to be used. Note that one Service can only handle IPv4 or IPv6, so if you want to use both protocols, you will need to create two separate services.

#### Listener Port

**File Configuration field:**

nListenPort

**Description:**

The port the `SETP Server` service listens on. The default value is 5432. This should be changed only if there is a definite need for it. Such a need typically arises from security concerns. If the port is changed, all reporting devices must also be configured to use the non-standard port. SETP operates over tcp.

#### Listener IP Address

**File Configuration field:**

szMyIPAddress

**Description:**

The `SETP Server` service can be bound to a specific IP address. You can either use an IPv4, an IPv6 Address, or a Hostname that resolves to an IPv4 or IPv6 Address. This feature is useful for multihome environments where you want to run different input services on different IP addresses. Please note that the default IP address `0.0.0.0` means any IP address.

#### Session Timeout

**File Configuration field:**

nTimeOutSession

**Description:**

This controls how long a session is to be opened from the server side.

### Options

### Enable SSL/TLS

**Note: if this Option is enabled, NON-SSL Clients will not be able to connect to this Service.**

**File Configuration field:**

nUseSSL

**Description:**

If this option is enabled then the `SETP Server` service accepts SSL / TLS setp connections only.

Please note: If this option is enabled, non-SSL SETP senders will not be able to connect.

### Use zLib Compression to compress the data

**File Configuration field:**

nZlibComp

**Description:**

When enabled, WinSyslog decompresses the zLib compressed data sent by the SETP senders. It is still be able to receive normal data. zLib compression is useful to reduce traffic in WAN environments.

### Notify Sender about Rule Action Errors?

**File Configuration field:**

bIndicateErrorToOrigin

**Description:**

Enable this option to communicate the outcome of an action back to the sender of the SETP message.

This communicates back the status of actions carried out on the receiver to the sender of the event. In essence, the sender system will know if the action failed or succeeded on the remote machine. It can then act exactly like the action was carried out on the local machine. The exact handling of failure states is depending on the event source.

An example: you have a machine running an EventLog Monitor and sending these events via SETP, and on the other side have all incoming events written into a database. If the database would be offline and the events not being written into it, the `SETP Server` service would return as the last message that the action failed (as long as this option is enabled) and generate an error event with ID 1005 (and generate a Success Event with ID 1012 if successful again). The sender would then halt and retry sending the event. This is because SETP is built somehow like TCP which ensures data transfer, but additionally can return a status to the sender if the following action was successful.

This happens because the Event Log Monitor (as well as the file monitor and others) is a restartable event source. It uses the outcome of actions to decide if the action is to be retried in another run of the same source. Other event sources have different behavior. The `Syslog server` service, for example, does not retry failed actions. This is due to the lossy nature of syslog, in which losing syslog messages is explicitly permitted (and favorable over taking up too many system resources by trying to buffer them).

Please Note: If you enable this feature, older WinSyslog Versions (4.2.x and below, as well as WinSyslog 7.2.x and EventReporter 8.2.x and below) may have trouble sending data over SETP once a Rule Exception occurs! If you intend to use this feature, make sure all WinSyslog Installations are at least Version 4.3.x (This applies for WinSyslog 7.3.x and EventReporter 8.3.x as well).

### RuleSet to use

**File Configuration field:**

szRuleSetName

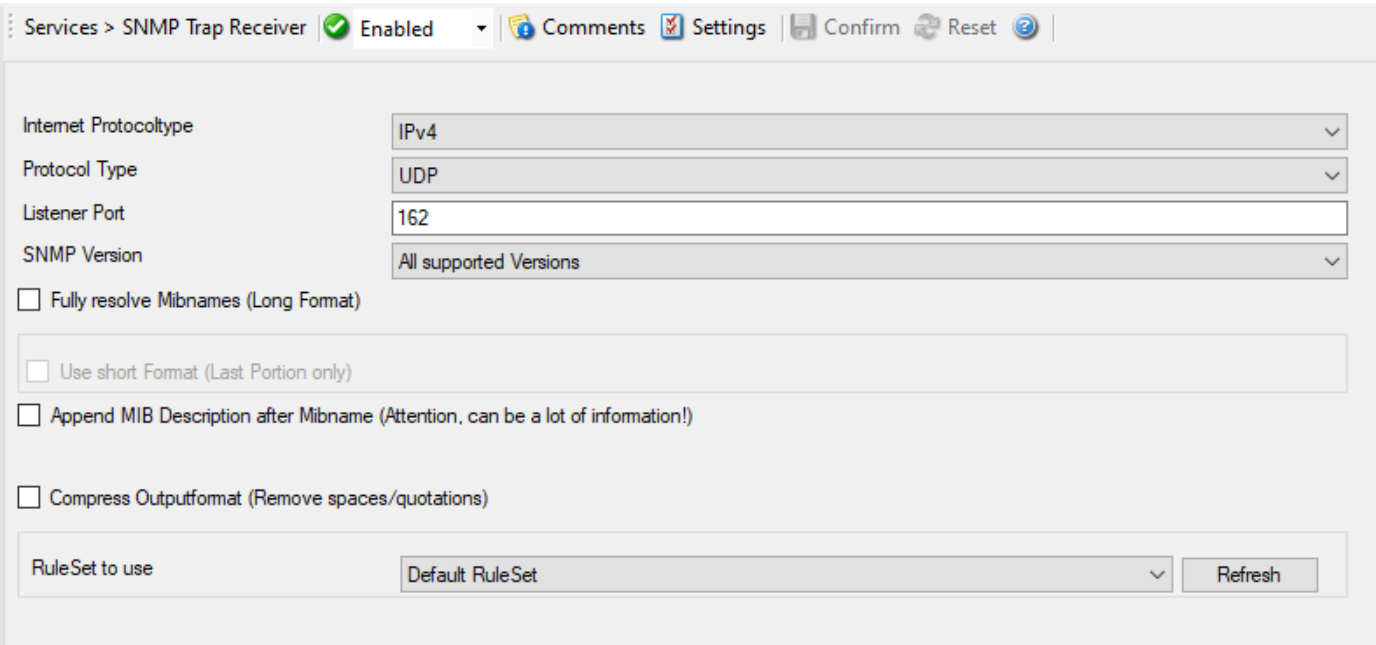
**Description:**

Name of the ruleset to be used for this service. The RuleSet name can be chosen from a drop-down list where you find your RuleSets.

## SNMP Trap Receiver

The `SNMP Trap Receiver` service receives SNMP messages. A trap is somewhat like a syslog message over another protocol (SNMP). It is generated by the device and contains the information that the device decides to transmit, plus a small set of standard items such as version and community.

The `SNMP Trap Receiver` service runs continuously based on the configuration mentioned below:




- Service - SNMP Trap Receiver*

### Internet Protocoltype

**File Configuration field:**

nInetType

**Description:**

Select the desired protocol type. IPv4 and IPv6 are available. The IPv6 protocol needs to be properly installed in order to be used. Note that one Service can only handle IPv4 or IPv6, so if you want to use both protocols, you will need to create two separate services.

### Protocol Type

**File Configuration field:**

nProtocolType

**Description:**

You can select whether the `SNMP Trap Receiver` service listens on UDP or TCP for SNMP traps.

### Listener Port

**File Configuration field:**

nPort

**Description:**

The port the `SNMP Trap Receiver` service listens on. If in doubt, leave it at the default of 162, which is the standard port for this.

### SNMP Version

**File Configuration field:**

nSnmpVersion

- -1 = All Supported Versions
- 0 = SNMP Version 1 only
- 1 = SNMP Version 2c only

**Description:**

Can be used to restrict the SNMP versions. The available values are:

- All Supported Versions (i.e. SNMP Version 1 and SNMP Version 2c only)
- SNMP Version 1 only
- SNMP Version 2c only

### Fully resolve MIB names (long format)

**File Configuration field:**

nResolveLongMibNames

**Description:**

This option fully resolves the MIB names like in the client MIB browser application.

### Use short format (last portion only)

**File Configuration field:**

nResolveMibNamesShort

**Description:**

Fully resolved MIB names including their tree can become very long and unreadable. Use this option to shorten them to the last portion of the full MIB name.

### Append MIB description after MIB name

**File Configuration field:**

nAddMibDescriptionToMsg

**Description:**

Append the MIB description after the MIB name. **Attention, this can be a lot of information.**

### Compress output format (remove spaces/quotations)

**File Configuration field:**

nCompressOutputFormat

**Description:**

When enabled the output format will be reduced to a minimum and comma separated. Here is a sample output:

```
source=127.0.0.1, community=public, version=Ver2,
iso.3.6.1.2.1.1.3.0=Timeticks: (3493305159) 404 days, 7:37:31.59,
iso.3.6.1.6.3.1.1.4.1.0=OID: iso.3.6.1.4.1.19406.1.2.2,
iso.3.6.1.4.1.19406.1.1.1.7=This is a SyslogTest
```

### General Values (Common settings for most services)

### RuleSet to Use

**File Configuration field:**

szRuleSetName

**Description:**

Name of the ruleset to be used for this service. The RuleSet name must be a valid RuleSet.

**Please Note:**

**Managing incoming traps works the same way as with the ``Syslog server`` service, for example.**

Incoming traps will be forwarded to the corresponding Ruleset and pass by rule after rule. There it can be filtered for general information like the “Community”, the “Version” or “Value” for example. Finally it will be processed by an action, which you can select to your needs. The SNMP Agent service will co-exist peacefully next to the Windows SNMP Agent and will not hinder it in its functionality. The Windows SNMP Agent listens to port 161, while WinSyslog and WinSyslog listen to port 162.**

**For internal processing, the variables of incoming SNMP messages will be added to a new property. Those properties will be named %snmp_var_x% with the x being a number starting with 1. You can use these custom properties for filtering and everywhere where you can use or print properties. For example, you can create a “send mail”-action. Here you can specify complete freely how the message will look like. You can use a introductory text and then let it show the error message in some context. This could look like this:**

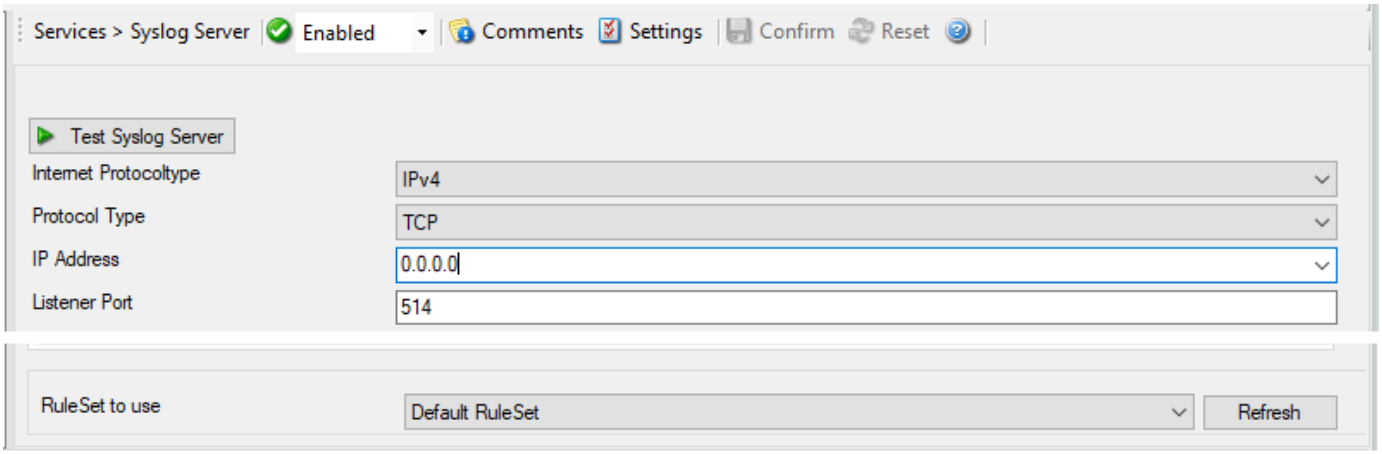
```
Hello Admin,
the following error occurred
%snmp_var_5%
Please take care at once.
Very urgent!
```

**The result will be, that the 5th property of the snmp trap will be inserted into the message text.**

### Syslog server

Configures the `Syslog server` service. In practice, this is the WinSyslog or WinSyslog input service that receives incoming syslog messages. Multiple protocols (IPv4/IPv6 and UDP/TCP) can be configured and are supported.

When configuring Syslog Services, the functionality can be checked using the Test Syslog server button. It will open the Syslog Test Message function from the configuration client.




- Service - Syslog server Global Properties*

#### Internet Protocoltype

**File Configuration field:**

nInetType

**Description:**

Select the desired protocol type. IPv4 and ipv6 are available. The IPv6 protocol needs to be properly installed in order to be used. **Note that one Service can only handle IPv4 or IPv6, so if you want to use both protocols, you will need to create two separate services.**

#### Protocol Type

**File Configuration field:**

nProtocolType

**Description:**

Syslog messages can be received via udp, tcp or rfc 3195 RAW. One service instance can only use one of the protocols at a time. Typically, syslog messages are received via UDP protocol, which is the default. The `Syslog server` service also can receive syslog messages via TCP and reliable syslog messages via TCP using the RFC 3195 RAW standard. Depending on which protocol type you choose, you get different option tabs. General and encoding are the same for everyone.

#### IP Address

**File Configuration field:**

szMyIPAddress

**Description:**

The `Syslog server` service can now be bound to a specific IP address. You can either use an IPv4, an IPv6 Address, or a Hostname that resolves to an IPv4 or IPv6 Address. This feature is useful for multihome environments where you want to run different syslog input services on different IP addresses. Please note that the default IP `address 0.0.0.0` means ANY IP Address.

#### Listener Port

**File Configuration field:**

nListenPort

**Description:**

The port the `Syslog server` service listens on. The typical (standard) value is 514. This should be changed only if there is a definite need for it. Such a need typically arises from security concerns. If the port is changed, all reporting devices (routers, printers …) must also be configured to use the non-standard port.

### RuleSet to use

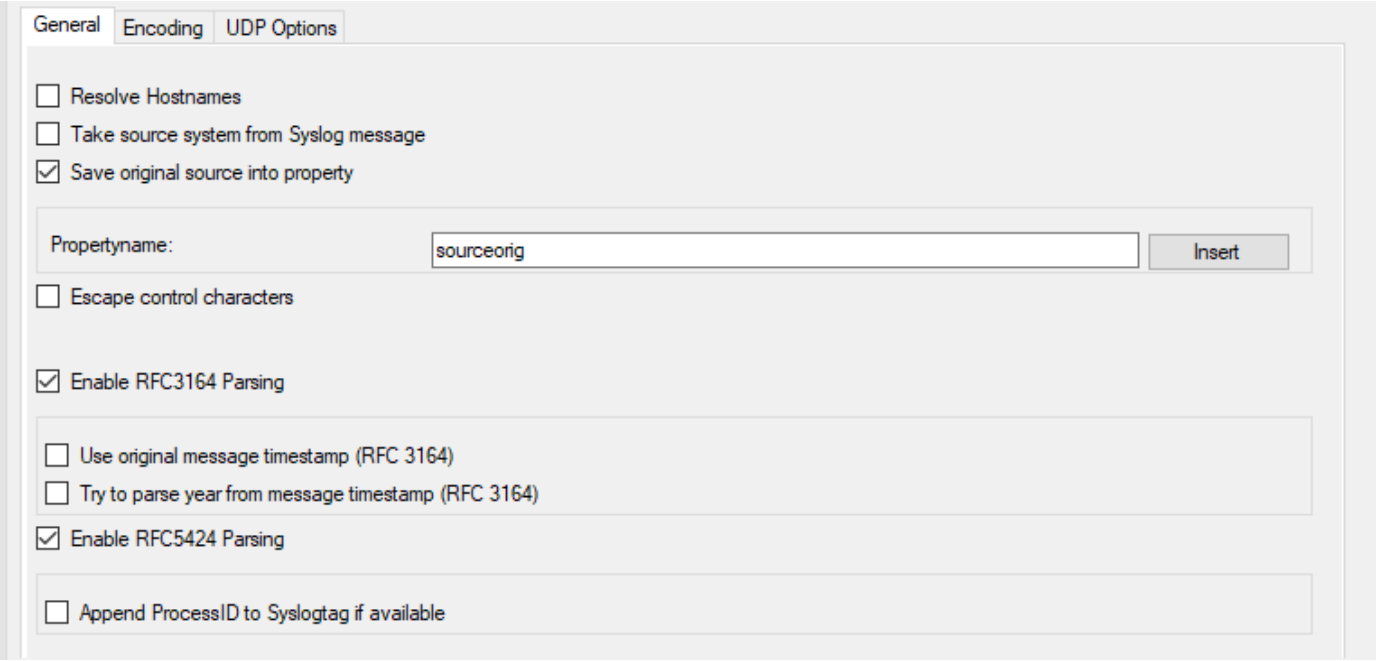
**File Configuration field:**

szRuleSetName

**Description:**

Name of the ruleset to be used for this service. The RuleSet name must be a valid RuleSet.

### General Options




- Service - Syslog server General Tab*

### Resolve Hostnames

**File Configuration field:**

nResolveNames

**Description:**

If this box is checked, the name of the source system is retrieved via DNS reserve name resolution. If unchecked, the IP address itself is used as the name.

**Please note that this setting does have any effect if the “Take source system from Syslog message” setting is checked. In this case, the message is always taken from the Syslog message itself.**

### Take source system from Syslog message

**File Configuration field:**

nTakeSourceSysFromSyslogMsg

**Description:**

If this box is checked, the name or IP address of the source system is retrieved from the Syslog message itself (according to rfc 3164). If left unchecked, it is generated based on the address, the message was received from.

**Please note that there are many devices, which do NOT generate RFC 3164 compliant messages. If you check this option here, you might see a very strange value as the event source!**

### Save original source into property

**File Configuration field:**

nSaveSourceIntoProperty

**Descriptions:**

When this options is enabled, the original network source will be stored into the custom defined property (%sourceorig% by default). In case the original network source is needed for filtering for example.

### Escape Control Characters

**File Configuration field:**

nEscapeControlCharacters

**Description:**

Control characters are special characters. They are used e.g. for tabulation, generating beeps and other non-printable uses. Typically, syslog messages should not contain control characters. If they do, control characters could eventually affect your logging. However, it might also be that control characters are needed.

With this setting, you can specify how control characters received should be handled. When checked, control characters are replaced by a 5-byte sequence with the ASCII character ID. For example, a beep is the ASCII BEL character. BEL is assigned the numerical code 7. So if a BEL is received, it would be converted to “<007>” inside your syslog message. When the box is left unchecked, no conversion takes place.

In any case, ASCII NULs are converted to “<000>” to prevent security issues in the log files.

**Please note: if you used double-byte character sets, control character escaping can cause your message to become clobbered. So be sure to leave it unchecked in that case.**

### Enable RFC3164 Parsing

**File Configuration field:**

nRFC3164Parsing

**Description:**

If this box is checked, rfc 3164 compliant message parsing is enabled. If unchecked, “traditional” Adiscon message parsing is selected. If you experience trouble with the sender host name or the timestamp, we suggest that you turn off RFC 3164 compliant message parsing. Many existing devices do not fully comply with RFC 3164 and this can cause those issues.

### Use Original Message Timestamp

**File Configuration field:**

nParseSyslogDate

**Description:**

If this box is checked, the timestamp is retrieved from the Syslog message itself (according to rfc 3164). If left unchecked, the timestamp is generated based on the local system time. The Syslog message timestamp does not contain time zone information. Thus, we strongly recommend unchecking this box if messages from devices in multiple time zones are to be received

### Try to parse year from message timestamp (RFC3164)

**File Configuration field:**

nRFC3164DetectYear

**Description:**

If enabled, the service will try to detect a Year after the usual RFC3164 Date Header.

### Enable RFC5424 Parsing

**File Configuration field:**

nRFC5424Parsing

**Description:**

If this box is checked, rfc 5424 compliant message parsing is enabled for Syslog RFC5424 Header detection and decoding. This also involves new usable Syslog properties. If unchecked, “traditional” Adiscon message parsing is selected. If you experience trouble with the sender host name or the timestamp, we suggest that you turn off RFC 5424 compliant message parsing. Many existing devices do not fully comply with RFC 5424 and this can cause those issues.

### Append ProcessID to SyslogTag if available

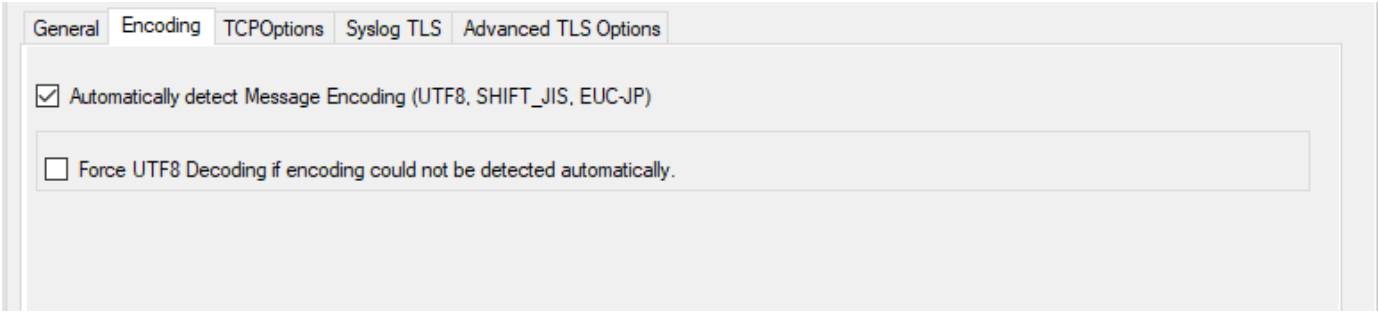
**File Configuration field:**

nRFC5424AddProcID2SyslogTag

**Description:**

This option is related to RFC5424 header parsing and was default in previous versions. However the default now is off in order to separate the Syslogtag from the ProcessID.

### Encoding Options




- Service - Syslog server Encoding Tab*

### Automatically detect Message Encoding (UTF8, SHIFT_JIS, EUCJP)

**File Configuration field:**

nTryDetectMessageEncoding

**Description:**

If enabled, the message will be checked for different encodings. This is important if you have syslog messages with multibyte characters. Once an encoding is detected, it will automatically be converted into UTF16 internally.

### Force UTF8 Decoding

**File Configuration field:**

nForceUTF8Decoding

**Description:**

This option forces UTF8 Decoding of all incoming messages. This is also useful for syslog messages encoded in UTF8 but missing the BOM within the Syslog message.

### UDP Options



- Service - Syslog server UDP Options Tab*

### Enable receiving from a UDP Multicast Group

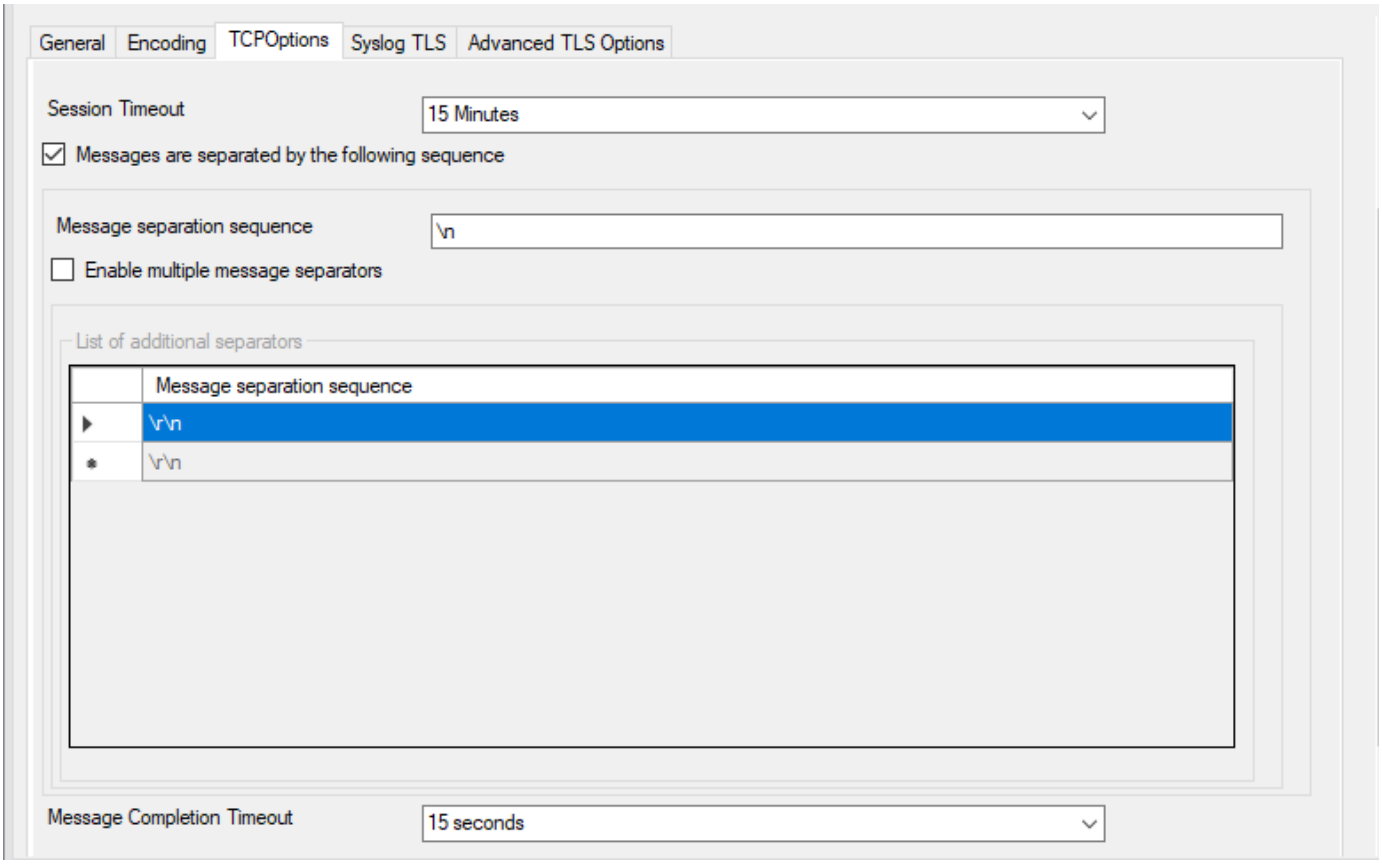
**File Configuration field:**

nEnableMultiCastGroup

**Description:**

This option supports receiving Syslog messages via multicast IP Addresses like `224.0.0.1` for example.

### TCP Specific Options




- Service - Syslog server TCP Options Tab*

### Session Timeout

**File Configuration field:**

nTimeOutSession

**Description:**

One of the TCP-specific options is the session timeout. This value declares, how long a TCP session may be kept open, after the last package of data has been sent. You can by default set values between 1 second and 1 day or you can use a custom value with a maximum of 2147483646 milliseconds. If you wish to disable the session timeout, you can use a custom value of 0 milliseconds to disable it.

### Maximum concurrent TCP sessions

**File Configuration field:**

nTCPMaxSessions

**Description:**

Limits how many TCP sessions this listener accepts at the same time. The default is `1024`. A value of `0` disables the cap. For internet-facing or untrusted networks, keep a finite limit so stalled connections cannot consume an unbounded number of receiver threads.

### TLS handshake timeout

**File Configuration field:**

nTLSHandshakeTimeout

**Description:**

Sets the timeout, in milliseconds, used while a new TLS connection completes its handshake. The default is `30000`. This setting applies before the normal TCP session timeout and helps disconnect clients that open a TLS socket but do not complete the handshake.

### Messages are separated by the following sequence

**File Configuration field:**

szMsgSep_[n]

**Description:**

If this option is checked, you can use multiple messages in the same transmission and the following options are enabled: Message separation sequence and Message Completion Timeout.

### Message separation sequence

**File Configuration field:**

nEnableTCPMsgSep

**Description:**

Determines, how you want to separate the messages. By default “rn” is the value for this, as most times a message ends with a carriage return and/or a line feed. But, you can choose your own separation sequence here as well.

### Enable multiple message separators

**File Configuration field:**

nEnableMultiTCPMsgSep

**Description:**

If you choose the checkbox you can use more than one message separator.

### Message Completion Timeout

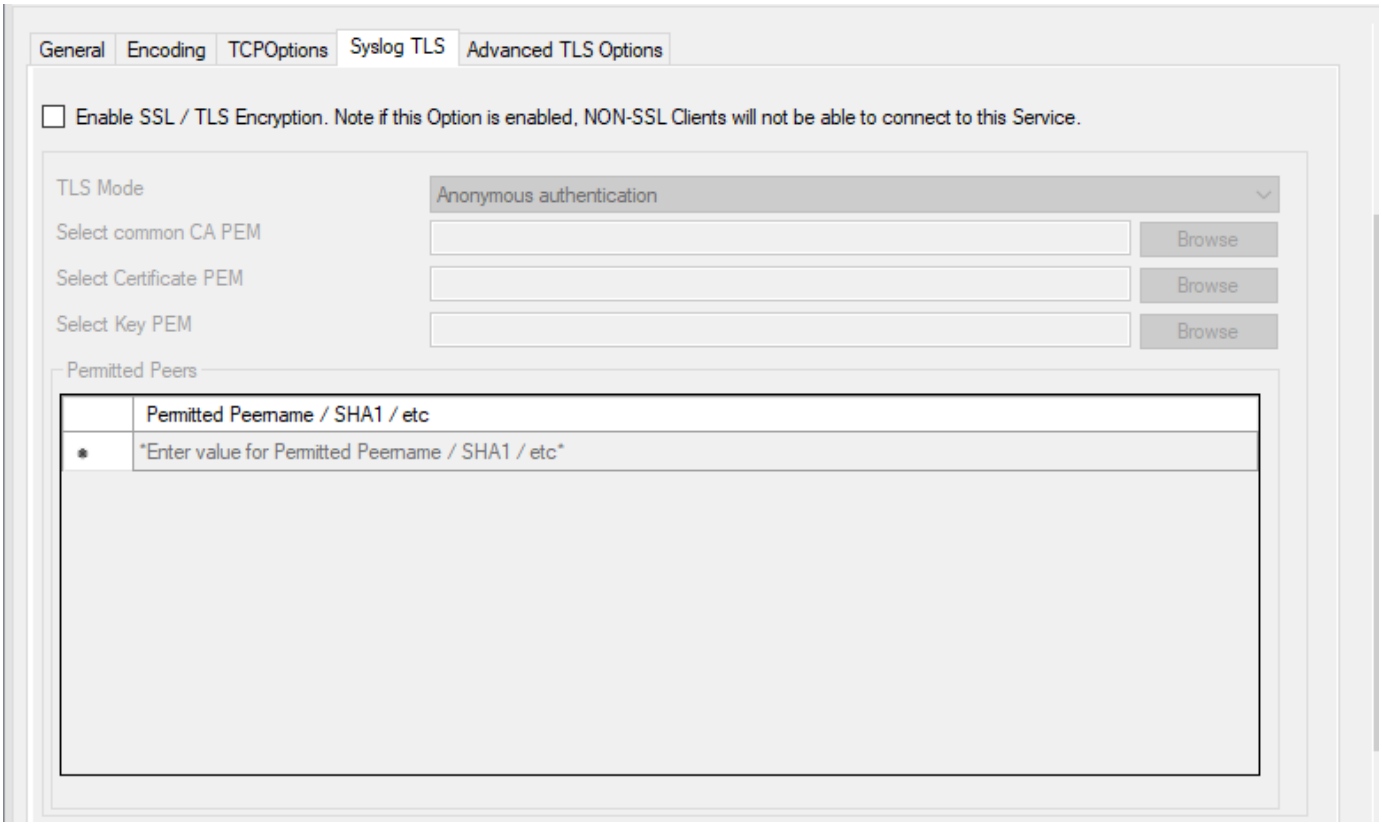
**File Configuration field:**

nTimeOutMsg

**Description:**

Here you can set the time that is allowed to complete a message. Is the time exceeded, but the message not yet completed, the rest will be treated as a new message. The counter is reset each time a new message begins. You can choose from multiple values between 1 second and 1 day, or choose a custom value in milliseconds (0 = disable, maximum = 2147483646)

### Syslog TLS




- Service - Syslog server Syslog TLS Tab*

### Enable SSL / TLS Encryption

**File Configuration field:**

nUseSSL

**Description:**

This option enables SSL / TLS encryption for your Syslog server. Please note, that with this option enabled, the server only accepts SSL / TLS enabled senders.

### TLS Mode

**File Configuration field:**

nTLSMode

**Description:**

The TLS mode can be set to the following:

**Anonymous authentication** Default option, which means any client certificate will be accepted, or even none. This provides encryption without peer authentication. Use it only when that is intentional for the deployment; use one of the x509 modes when peer authenticity matters.

**x509/name (certificate validation and name authentication)** When this mode is selected, the subject within the client certificate will be checked against the permitted peers list. This means the Syslog server will only accept the secured connection if it finds the permitted peer in the subject. Newer hardened configurations can disable the legacy subject substring match and rely on certificate name checks instead.

**x509/fingerprint (certificate fingerprint authentication)** This mode creates a SHA1 Fingerprint from the client certificate it receives, and compares it to fingerprints from the permitted peers list. You can use the debuglog to see fingerprints of client certificates which were not permitted. SHA-1 fingerprint labels remain supported for RFC 5425 interoperability. Hardened configurations can disable legacy substring matching and use exact normalized SHA-1 or SHA-256 fingerprints.

**x509/certvalid (certificate validation only)** A Syslog Sender is accepted when the client certificate is valid. No further checks are done.

### Select common CA PEM

**File Configuration field:**

szTLSCAFile

**Description:**

Select the CA certificate or CA bundle used to validate certificates presented by connecting clients. If you use a CA chain, include the intermediate CA certificates first and the root CA certificate last.

### Select Certificate PEM

**File Configuration field:**

szTLSCertFile

**Description:**

Select the server certificate in PEM format. This is the certificate the Syslog server presents to connecting clients. If needed, append the intermediate CA certificates after the server certificate so clients can validate the chain.

### Select Key PEM

**File Configuration field:**

szTLSKeyFile

**Description:**

Select the private key in PEM format that matches the server certificate. Passphrase-protected private keys are not supported.

### Permitted Peers

### Permitted Peername / SHA1 / etc.

**File Configuration field:**

szIP_[n]

**Description:**

This list contains all permitted peers. If x509/name is used, this can contain parts of the client certificate subject. For example if you have CN = secure.syslog.msg in the certificate subject, you can add “secure.syslog.msg” as permitted peer. When using x509/fingerprint, this list holds a list of permitted SHA1 fingerprints. The fingerprints can either be generated with OpenSSL Tools or grabbed from the debug logfile. The format is like described in RFC 5425, for example: `SHA1:2C:CA:F9:19:B8:F5:6C:37:BF:30:59:64:D5:9A:8A:B2:79:9D:77:A0`.

### TLS hardening compatibility options

**File Configuration fields:**

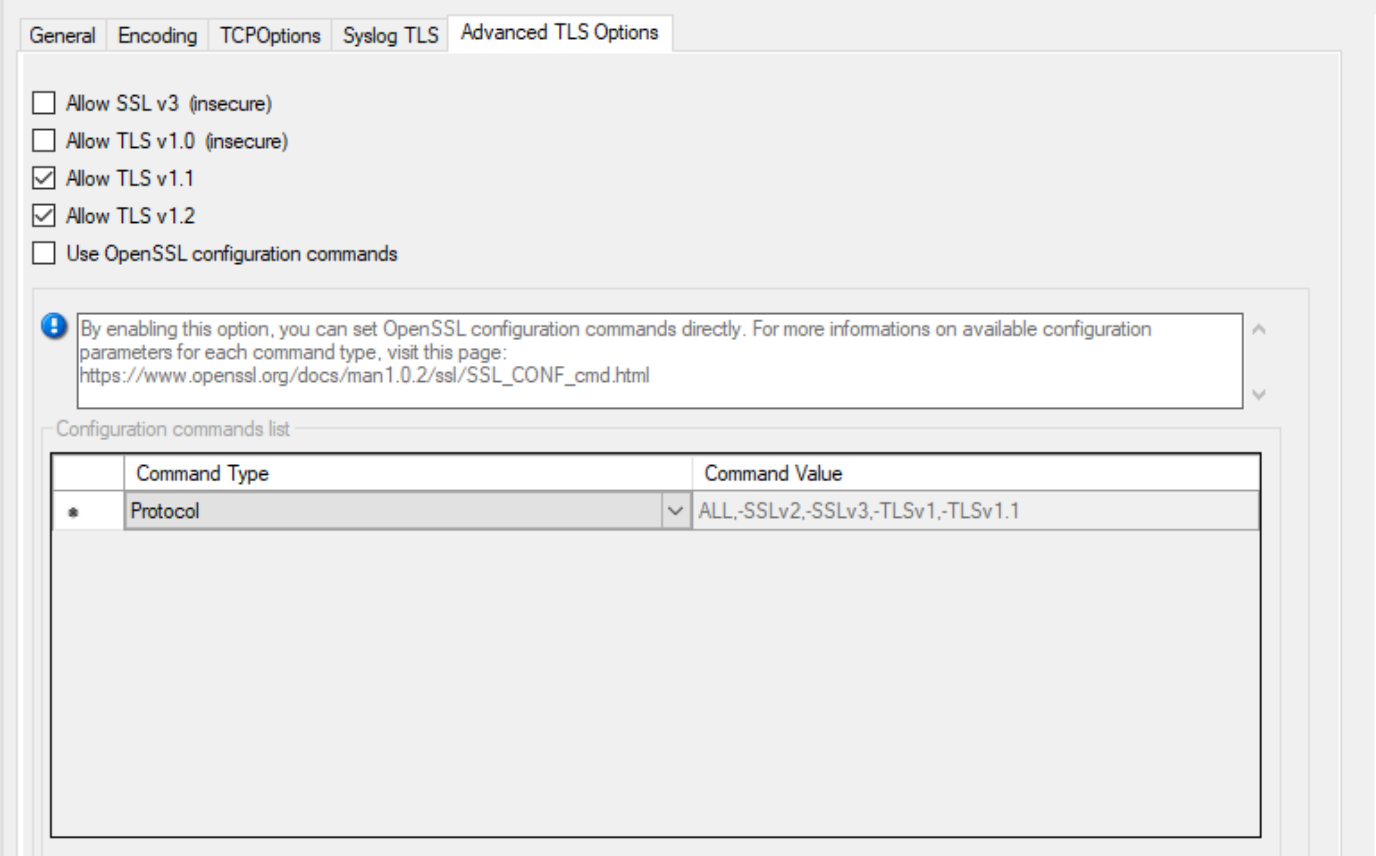
nTLSAllowExpiredCerts, nTLSLegacyPeerNameMatch, nTLSLegacyFingerprintMatch

**Description:**

These compatibility options control stricter certificate handling for authenticated TLS modes. The defaults preserve legacy behavior: `nTLSAllowExpiredCerts=1` accepts expired client certificates, `nTLSLegacyPeerNameMatch=1` allows the historic subject substring match for x509/name mode, and `nTLSLegacyFingerprintMatch=1` allows the historic SHA-1 fingerprint substring match.

Recommended hardened values are `0` for all three settings. With those values, expired client certificates are rejected, x509/name mode uses certificate name matching, and x509/fingerprint mode requires exact normalized SHA-1 or SHA-256 fingerprints.

### Advanced TLS




- Service - Syslog server Advanced TLS Options Tab*

### Allow SSL v3

**File Configuration field:**

nTLSAllowSSLv3

**Description:**

This option enables insecure protocol method SSLv3. We recommend NOT enabling this option as SSLv3 is considered broken.

### Allow SSL v1.0

**File Configuration field:**

nTLSAllowTLS10

**Description:**

This option enables insecure protocol method TLSv1. We recommend NOT enabling this option as TLSv1 is considered broken.

### Allow SSL v1.1

**File Configuration field:**

nTLSAllowTLS11

**Description:**

This option enables protocol method TLS1.1 which is enabled by default.

### Allow SSL v1.2

**File Configuration field:**

nTLSAllowTLS12

**Description:**

This option enables protocol method TLS1.2 which is enabled by default.

### Use OpenSSL configuration commands

**File Configuration field:**

nTLSUseConfigurationCommands

**Description:**

By enabling this option, you can set OpenSSL configuration commands directly. For more information's on available configuration parameters for each command type, visit this page: https://www.openssl.org/docs/man1.0.2/ssl/SSL_CONF_cmd.html

We allow to the set the following OpenSSL configuration commands in the configuration commands list.

- CipherSuite: This sets the available ciphers for TLS >= v1.3. For TLS < v1.3 use Ciphers instead. Note: setting this option will OVERWRITE the internal default CipherSuite.
- Ciphers: This sets the available ciphers for TLS < v1.3. For TLS >= v1.3 use CipherSuite instead. Setting this option will OVERWRITE the internal default cipher list.
- CipherString: Sets the allowed/disallowed used Ciphers. Setting this value will OVERWRITE the internal default ciphers.
- SignatureAlgorithms: This sets the supported signature algorithms for TLS v1.2.
- Curves: This sets the supported elliptic curves.
- Protocol: Sets the supported versions of the SSL or TLS protocol. This will OVERWRITE the Allow SSL options from above!
- Options: The value argument is a comma separated list of various flags to set.

### Allow TLS v1.3

**File Configuration field:**

nTLSAllowTLS13

**Description:**

This option enables protocol method TLS1.3 which provides enhanced security and performance.

**When setting advanced configuration commands, we highly recommend to enable**

debug logging and review it after changes have been made. An error will be logged in the debug logfile if a configuration command cannot be processed successfully.

### Fail on TLS configuration command error

**File Configuration field:**

nTLSFailOnConfigCommandError

**Description:**

Controls whether the listener fails to start TLS when an OpenSSL configuration command cannot be applied. The default is `0`, which keeps compatibility by logging the error and continuing. Set this value to `1` for hardened deployments where a weaker-than-intended TLS configuration should stop the listener instead of being accepted.

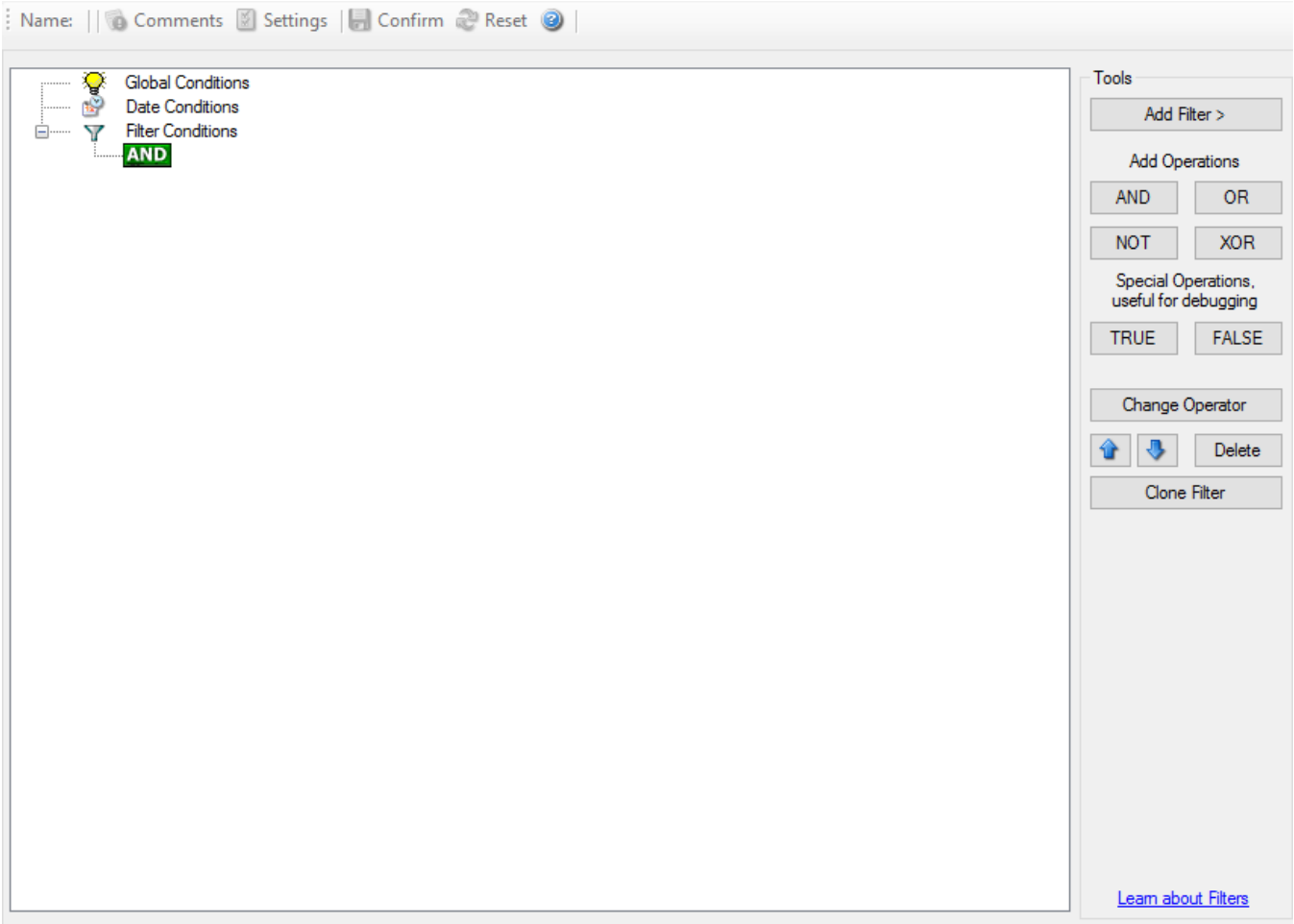
## Filter conditions

Filter conditions determine whether a rule matches an event. If a condition evaluates to true, the actions in that rule run.

WinSyslog supports simple filters and complex Boolean trees. This lets you use broad capture rules for storage or narrow, high-signal rules for alerting and forwarding.

## Default filter behavior

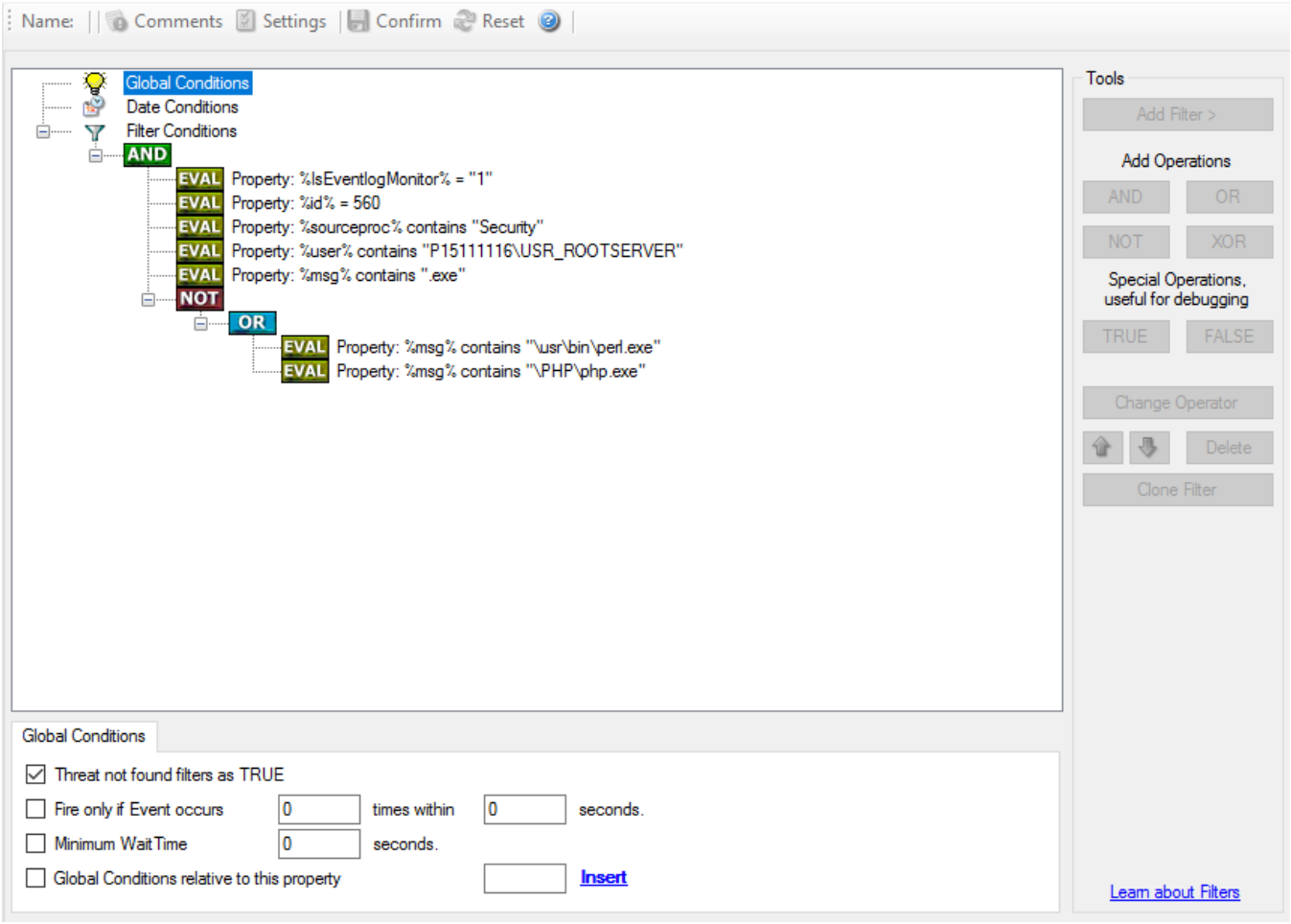
By default, the filter-condition tree contains a single top-level AND. That structure always evaluates to true until you add real conditions, so it is commonly used for broad capture rules such as writing all incoming events to a file or database.




*Default filter tree with a single top-level AND condition.*

## Building more selective filters

More advanced rules can combine multiple conditions with nested Boolean logic. The sample below shows a more selective filter structure.




*Example of a more selective filter with nested Boolean conditions.*

## How to think about filter logic

A useful mental model is:

- a filter that evaluates to **true** allows the rule's actions to run
- a filter that evaluates to **false** prevents those actions from running
- negation with **NOT** is often the key to expressing exceptions cleanly

The classic example is exclusion logic: if a condition should match everything except a small set of known-safe events, define the safe events first and then negate that result.

## Practical guidance

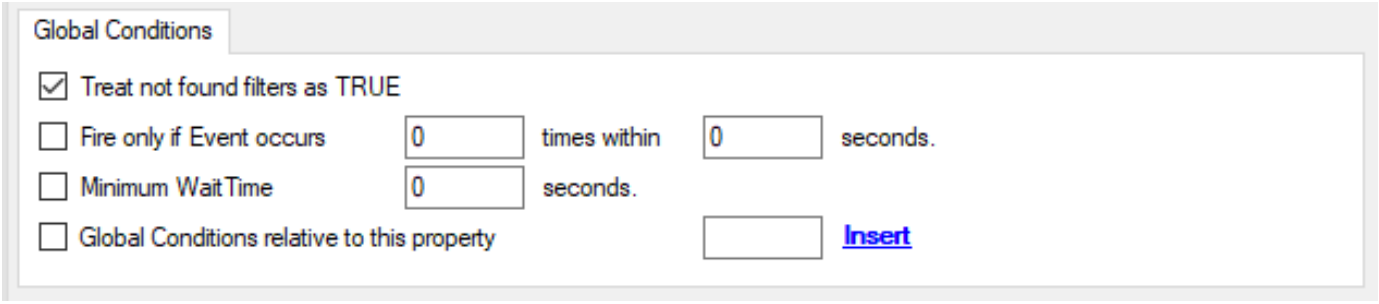
- String comparisons in filter conditions are case-sensitive.
- Start with the simplest condition that solves the problem.
- Add nesting only when simpler rule separation would not be clearer.
- Use wait times and throttling carefully to prevent alert storms.

If you need advanced modeling help, see complex filter conditions.

## Filter condition reference pages

## Global Conditions

Global Conditions apply to the rule as whole. They are automatically combined with a logical “AND” with the conditions in the filter tree.




- Global Conditions*

**Treat not found Filters as TRUE**

If a property queried in a filter condition is not present in the event, the respective condition normally returns “FALSE”. However, there might be situations where you would prefer if the rule engine would evaluate this to “TRUE” instead. With this option, you can select the intended behavior. If you check it, conditions with properties not found in the event evaluates to “TRUE.

**Fire only if Event occurs**

This is kind of the opposite of the “Minimum WaitTime”. Here, multiple events must come in before a rule fires. For example, this time we use a ping probe. Ping is not a very reliable protocol, so a single ping might be lost. Thus, it may not be the best idea to restart some processes just because a single ping failed. It would be much better to wait for repetitive pings to fail before doing so.

Exactly this is why the “Fire only if Event Occurs” filter condition is made for. It waits until a configured amount of the same events occurs within a period. Only if the count is reached, the filter condition matches and the rule can fire.

**Note: If you used previous versions of the product, you might remember a filter called “Occurrences”. This has just been renamed.**

**Minimum Wait Time**

This filter condition can be used to prevent rules from firing too often. For example, a rule might be created to check the status of a port probe event. The port probe probes an smtp server. If the event is fired and the rule detects it, it spawns a process that tries to restart the service. This process takes some time. Maybe the SMTP gateway need some more time to fully start up so that the port probe might fail again while the problem is already taken care of. The port probe as such generates an additional event.

Setting a minimum wait time prevents this second port probe event to fire again if it is – let’s say – within 5 minutes from the original one. In this case, the minimum wait time is not yet reached and as such, the rule does not match. If, however, the same event is generated 5 hours later (with the mail gateway failing again), the rule once again fires and corrective actions are taken.

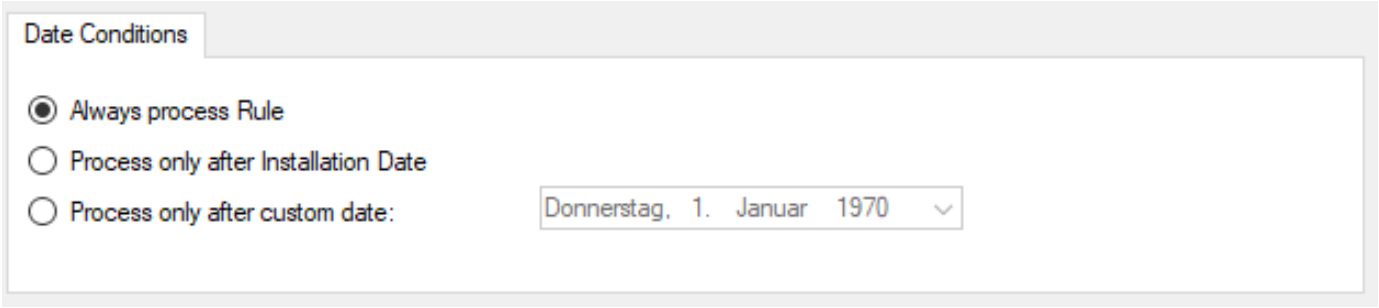
**Global Conditions relative to this property**

This feature enables you to control the Global Conditions based on a property.

For example take the source of a message as property. In this case, the Minimum WaitTime for example would be applied individual on each message source.

### Date Conditions

Rule processing can be bound to a specific or the installation date. By default a Rule will always be processed.




- Date Conditions*

**Always process Rule**

No date filter will be applied

**Process only after Installation Date**

Rule will only be processed if message was generated after the application installation date.

**Process only after custom date**

Rule will only be processed if message was generated after the custom specified date.

### Operators

In general, operators describes how filter conditions are linked together. The following operators can be used. Boolean operators like “AND” or “OR” can be used to create complex filter conditions.

**AND:**

All filters placed below must be true. Only then AND returns TRUE.

**OR:**

If one or both of the filters placed below is true, OR returns TRUE.

**NOT:**

Only one Filter can be placed below NOT operator, and if the filter evaluation is true, NOT returns FALSE.

**XOR:**

If one of the two filters are possible in the XOR Operator.

**TRUE:**

Useful for debugging, just returns TRUE.

**FALSE:**

Useful for debugging as well, returns FALSE.

## Filters

Filters can be added under each Operation node. There are a few common filters which can be used for all services and there are special filters which only apply if a special kind of Information Unit is evaluated.

**What happens with Filters that are not available in an “Information Unit”?**

Every filter that is not found in an Information Unit is ignored in the filtering process. If you want to create filters specialized for types of Information Units, always make sure to add an “Information Unit Type” filter.

An example, you have one ruleset, rule and action. In the filters you have one EventID filter. Then you have two services, one Eventlog Monitor and the other is Heartbeat monitor both pointing to this ruleset. The Information Units from the Eventlog Monitor would be filtered correctly, but those from the Heartbeat monitor would not be filtered as they do not have an EventID property. The EventID filter would be ignored and the actions would be executed every time.

**Note, if a filter is used that does not apply to the evaluated Info Unit, it will be just ignored. This gives you the possibility to build one filter set for several types of Information Units.**

There are different types of filters, and so there a different ways in which you can compare them to a value. The following Types exist:

**String:**

Can be compared to another String with “=”, “Not =”, “Range Match” or through

### REGEX Compare Operation

The property will be evaluated against a regular expression. Everything known in the regular expression syntax can be used to define a matching pattern.

Here are some regular expressions samples:

Regular Expression:

```
[0-9]{4,4}-[0-9]{1,2}-[0-9]{1,2} [0-9]{1,2}:[0-9]{1,2}:[0-9]{1,2}
```

Matches typical Date like `2018-11-20 12:11:01`

Regular Expression: `\n[0-9]{4,4}`

Matches Linefeed and 4-digit number.

Regular Expression: `(;|:)` Matches semicolon or a colon.

More samples and details about the Regular Expression Syntax can be found here:

https://msdn.microsoft.com/en-us/library/bb982727(v=vs.90).aspx

**number:**

can be compared with another number with “=”, “not =”, “<” and “>”

**boolean:**

can be compared to either true or false with “=” and “not =”

**time:**

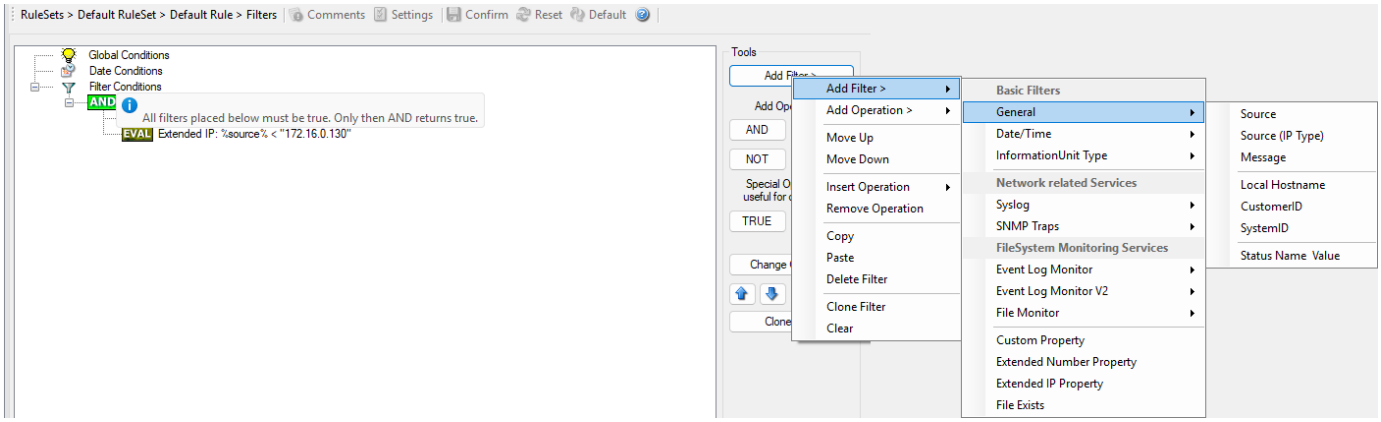
can be compared with another time but only with “=”

the list of possible filters, which can be evaluated is described in the following sections.

## Basic filters

## General

These are non-event log specific settings.




*WinSyslog Filter Conditions - General*

**Source**

This filter condition checks the system that generated the information unit. For example, in case of the Syslog server, this is the Syslog device sending a Syslog message.

This filter is of type string and should contain the source system name or IP address.

**Source System (IP)**

The IP Filter can basically work on any property, but we recommend to only use it on the %source% property, as we usually can be sure that this is contains a valid IP Address or hostname. The IP Filter can filter against hostnames and IP Addresses, hostnames are automatically resolved using the internal DNSCache (for obvious performance reasons).

This filter is of type string and should contain the source system name or IP address.

Please see the description for extended ip property for more information on how to use this property.

**Message Content**

The message content filter condition is very powerful. It evaluates to true if the specified content is found anywhere within the message. As there is implicit wildcarding, there is no need for extra wildcards to be specified.

The content search can be limited to a region within the message. To do so, select a starting and ending position within the string by choosing the **“contains within range”** compare operation. This can be done by specifying the start range and end range into the respective boxes.**Please note that you can enter the character position you desire in these fields. The default “Start Range” and “End Range” are set to 0.**

If you would like to search for a string just between positions 10 and 50, specify these values as start and end values, respectively. Similarly if you want to receive all logs from 192.168.0.1 then set this as:

- Property value = 192.168.0.0
- Range Start = 0
- Range End = 10

Which means 10 characters starting at zero (“192.168.0.”). Please note that the final DOT must be included. If you just used range “9”, then 192.168.010 would also match.

This filter is of type string.

**CustomerID**

CustomerID is of type integer provided for customer ease. For example if someone monitors his customer’s server, he can put in different CustomerIDs into each of the agents. Let us say someone monitors servers A and B. A has 5 servers all of them with CustomerID = 1 and B has 2 servers all of them with CustomerID = 2. Both A and B happen to have a server named “SERVER”. Together with the customerID, these machines are now uniquely identifiable. This is user configurable.

CustomerID (Type=Number).

**SystemID**

SystemID is of type integer to be used by our customer. In addition, it is user configurable.

SystemID (Type=Number).

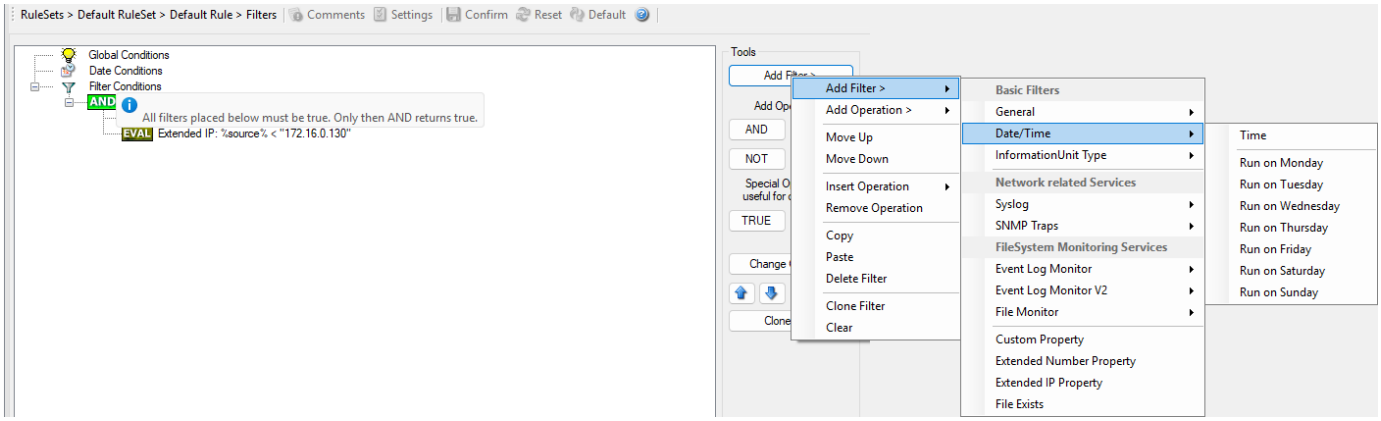
**Status Name and Value**

These filter type corresponds to set status action .

Status Name and Value (Type=String)

## Date/Time

This filter condition is used to check the time frame and / or day of week in which an event occurred.




*WinSyslog Filter Conditions - Date/Time*

**Time**

This filter condition is used to check the period in which an event occurred. For example, a Syslog message from a Cisco router saying that it dialed up is normal if it occurs during office hours. If it occurs at night, so, it is an alerting signal and an administrator might receive notification of this event (while he might otherwise decide to discard it). This can be done with the time setting.

You can also set the timezone setting (DefaultTimemode, UTC or Localtime) for the TimeMode's (DeviceReportedTime/ReceivedTime).

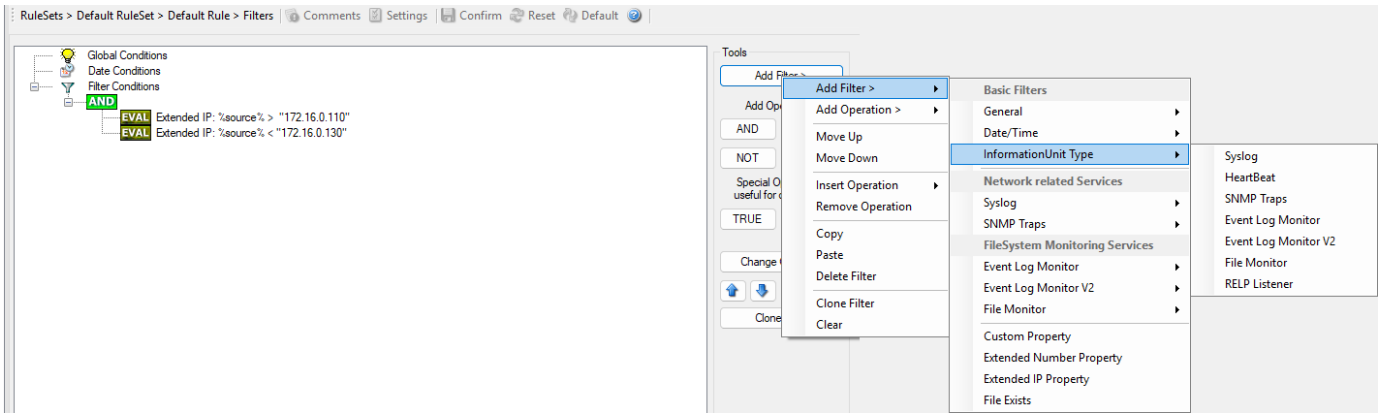
**Weekdays**

This is closely equivalent to the time filter condition, except that it is applied on a per-day basis. So it can be used to detect for example events occurring on weekends and act differently on them. The following filters are available:

1. Run on Monday (Type=Boolean)
2. Run on Tuesday (Type=Boolean)
3. Run on Wednesday (Type=Boolean)
4. Run on Thursday (Type=Boolean)
5. Run on Friday (Type=Boolean)
6. Run on Saturday (Type=Boolean)
7. Run on Sunday (Type=Boolean)

### InformationUnit Type

Select the specific information if a rule should just be processed for some information unit types. This is especially useful if a specific type needs non-standard processing. There is one pre-defined filter for each possible InformationUnit Type available (shown below).




*WinSyslog Filter Conditions - InformationUnit Type*

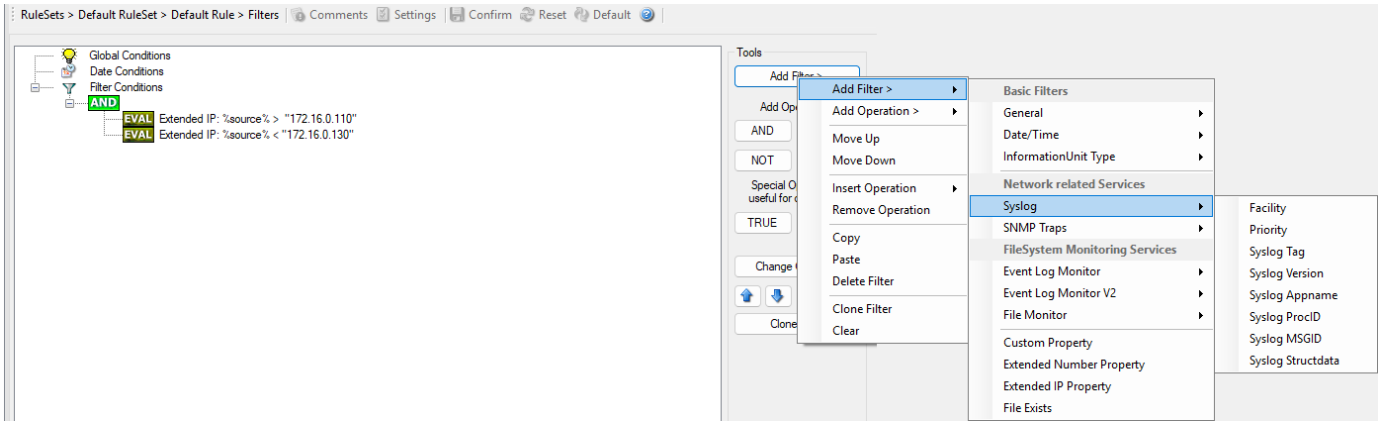
The following filters are available:

1. Syslog (Type=Boolean)
2. Heartbeat (Type=Boolean)
3. SNMP Traps (Type=Boolean)
4. Event Log Monitor (Type=Boolean)
5. File Monitor (Type=Boolean)
6. Ping Probe (Type=Boolean)
7. Port Probe (Type=Boolean)
8. NT Services Monitor (Type=Boolean)
9. Disk Space Monitor (Type=Boolean)
10. Database Monitor (Type=Boolean)
11. Serial Port Monitor (Type=Boolean)
12. CPU/Memory Monitor (Type=Boolean)
13. FTP Probe (Type=Boolean)
14. HTTP Probe (Type=Boolean)
15. IMAP Probe (Type=Boolean)
16. NNTP Probe (Type=Boolean)
17. POP3 Probe (Type=Boolean)
18. SMTP Probe (Type=Boolean)

### Network-related filters

## Syslog

Syslog related filters are grouped here. Please keep in mind that every Information Unit has assigned a Syslog priority and facility and thus these filters can be used with all Information Units.




*WinSyslog Filter Conditions - Syslog*

### Syslog Facility

The information unit must have the specified Syslog facility value. For Syslog type information units, it is the actual Syslog priority code, for all others it is a value mapped on a best effort basis.

This filter is of type number.

### Syslog Priority

The information unit must have the specified Syslog priority value. For Syslog type information units, it is the actual Syslog priority code, for all others it is a value mapped on a best effort basis.

The first list box allows to set a matching mode. The operations “less than” (<), “greater than” (>), and “equal” (=) can be selected. The match is made depending on these operations, so a “less than” operation means that all priorities below the specified priority match. Please note that the specified priority is not a match. If you would like to include it, be sure to specify the next higher one.

This filter is of type number.

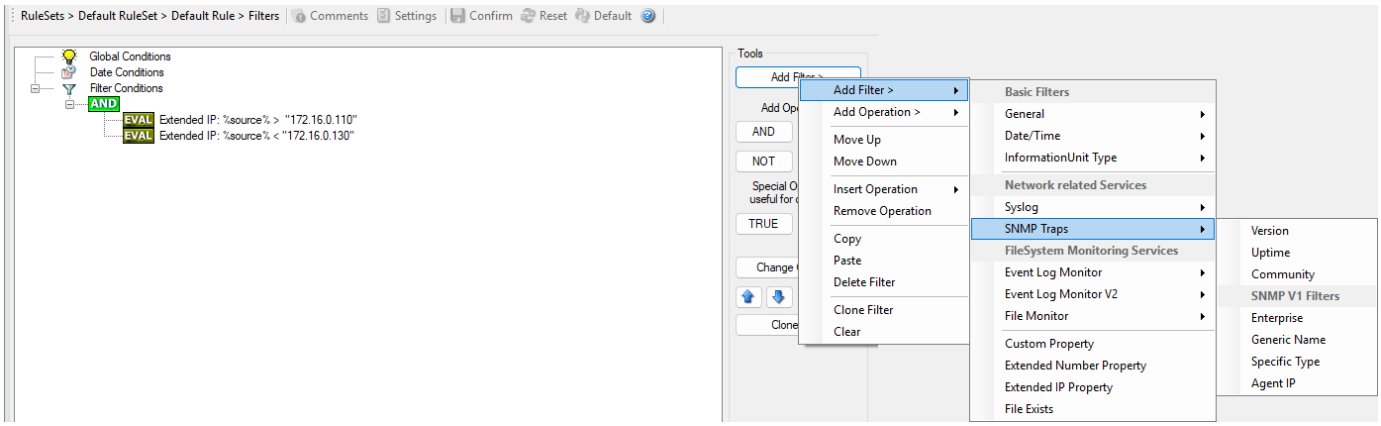
### Syslog Tag

This filter is of type string.

## SNMP Traps

Using SNMP Traps, since WinSyslog 3.0 now can be used to manage and monitor all sorts of equipment including computers, routers, wiring hubs, toasters, and jukeboxes.

A trap is generated when the device feels it should do so and it contains the information that the device feels should be transmitted.




*WinSyslog Filter Conditions - SNMP Traps*

**Community**

It corresponds to the respective SNMP entity.

This filter is of type string.

**Enterprise**

It corresponds to the respective SNMP entity.

This filter is of type string.

**Generic name**

It corresponds to the respective SNMP entity.

This filter is of type string.

**Version**

It corresponds to the respective SNMP entity.

This filter is of type number.

**Uptime**

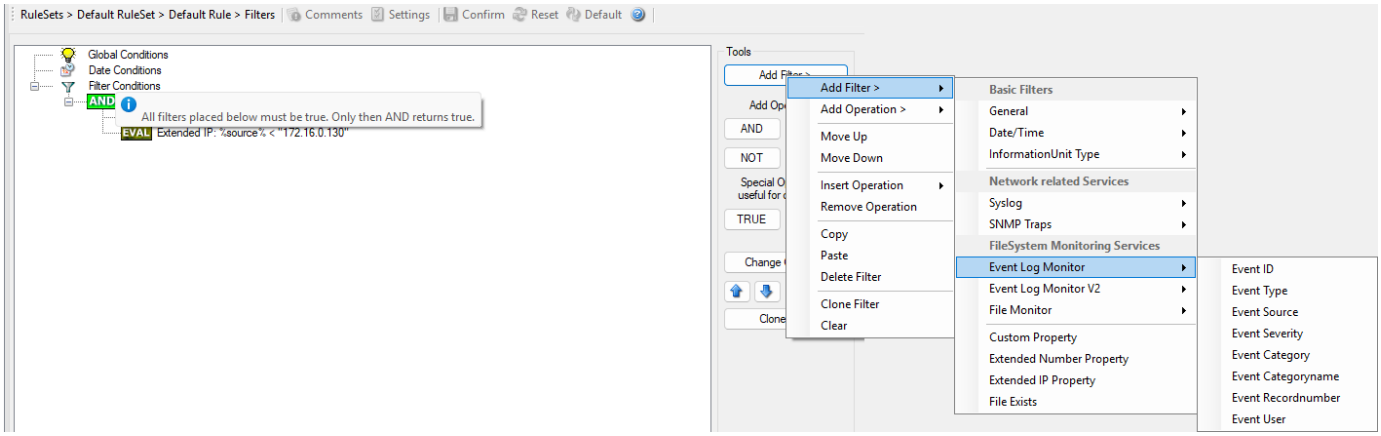
It corresponds to the respective SNMP entity.

This filter is of type string.

## Event and file monitoring filters

## Event Log Monitor

Event Log Monitor specific filters are grouped here.




*WinSyslog Filter Conditions - Event Log Monitor V1*

**Event ID**

This is the event log ID as specified in the Windows Event Log. If enabled, the event must have the configured event ID or the rule will not match. This is an integer value.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type number.

**Event Type**

This is the event log type as specified in the Windows Event Log. If enabled, the event must have the configured event type or the rule will not match. The supported values can be selected from the list box.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

**Event Source**

This is the event log source as specified in the Windows Event Log. If enabled, the event must have the configured event source or the rule will not match. This is a string value. There must be an exact match. Please note that this value is case-sensitive.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

**Event Severity**

This is the event log severity as specified in the Windows Event Log. If enabled, the event must have the configured severity or the rule will not match. The supported values can be selected from the list box.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type number.

**Event Category**

This is the event log category as specified in the Windows Event Log. If enabled, the event must have the configured event category or the rule will not match.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type number.

**Event Categoryname**

This value contains the Category value as string if it can be resolved. Otherwise it contains the category number.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

**Event Recordnumber**

This value contains the internal event record number. Please note that if the event log has been truncated before, it may not start with 0 or 1 but a higher number.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type number.

**Event User**

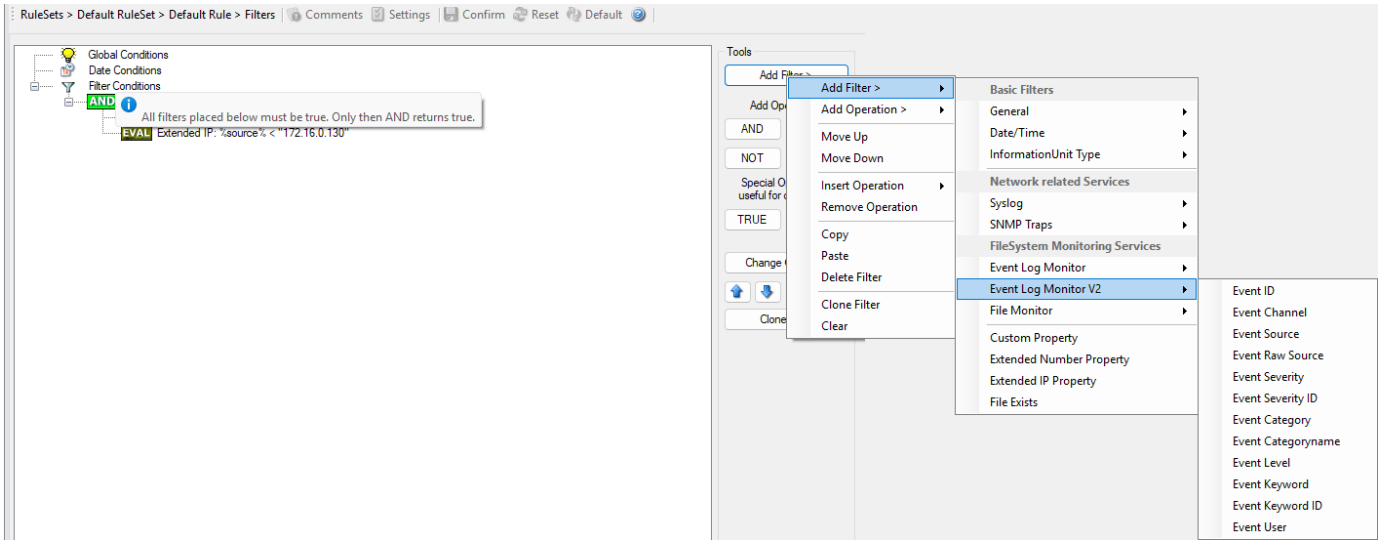
This is the event log user as specified in the Windows Event Log. If enabled, the event must have the configured event user or the rule will not match. Since it is a string value there must be an exact match. Please note that this value is case-sensitive.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

## Event Log Monitor V2

Event Log Monitor V2 specific filters are grouped here.




*WinSyslog Filter Conditions - Event Log Monitor V2*

**Event Channel**

The channel property for event log entries, for classic Event logs they match the %nteventlogtype% property, for new event logs, they match the “Event Channel”. If enabled, the event must have the configured event type or the rule will not match. There must be an exact match. Please note that this value is case-sensitive.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

**Event Raw Source**

This contains the full internal name of the event source for new event logs, for classic event logs it contains the same value as in %sourceproc%. If enabled, the event must have the configured event source or the rule will not match. This is a string value. There must be an exact match. Please note that this value is case-sensitive.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

**Event SeverityID**

This is the internal ID of the event log level as number. This is a integer value. There must be an exact match. Please note that this value is case-sensitive.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type number.

**Event Level**

This is a textual representation of the event log level (which is stored as number in %severityid%). This property is automatically localized by the system. If enabled, the event must have the configured level or the rule will not match. There must be an exact match. Please note that this value is case-sensitive.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

**Event Keyword**

This is a textual representation of the event keyword. This property is automatically localized by the system. If enabled, the event must have the configured event keyword or the rule will not match. There must be an exact match. Please note that this value is case-sensitive.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

**Event KeywordID**

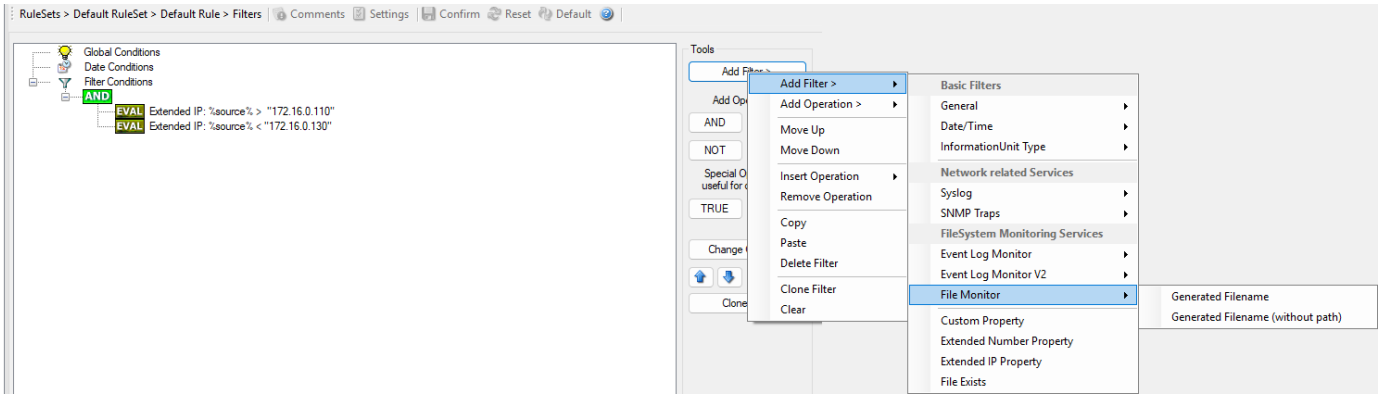
This is the internal keyword ID as string. If enabled, the event must have the configured event keyword ID or the rule will not match. There must be an exact match. Please note that this value is case-sensitive.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

## File Monitor

File Monitor specific filter is described here.




*WinSyslog Filter Conditions - File Monitor*

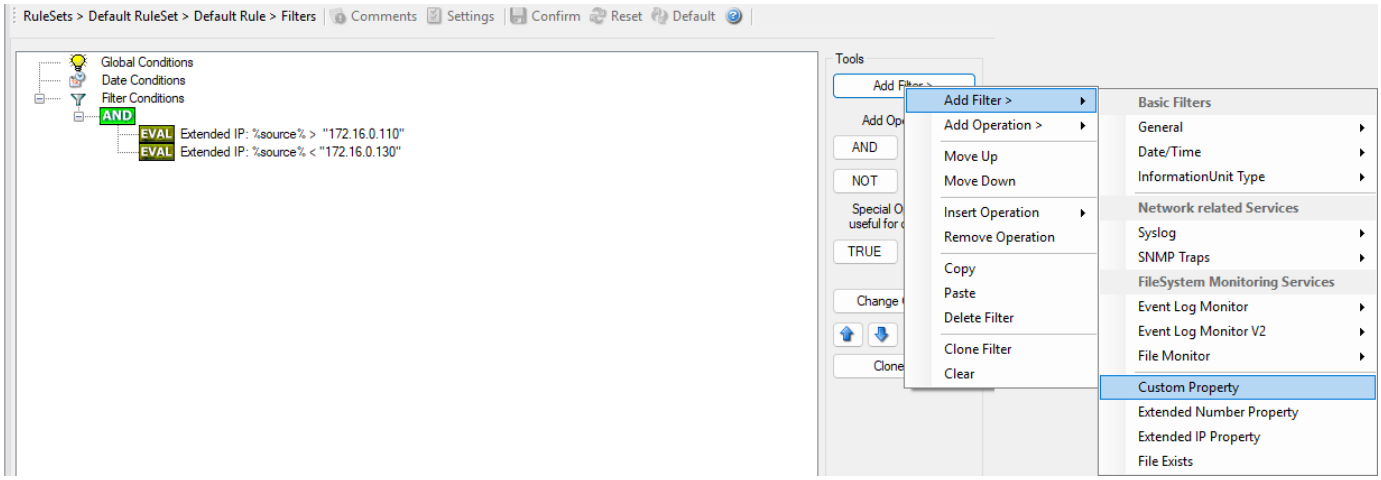
**Generated Filename**

The configured generic name of the file being reported. Filter has to match exactly to work.

## Custom property filters

## Custom Property

Custom Property specific filter is described here.




*WinSyslog Filter Conditions - Custom Property*

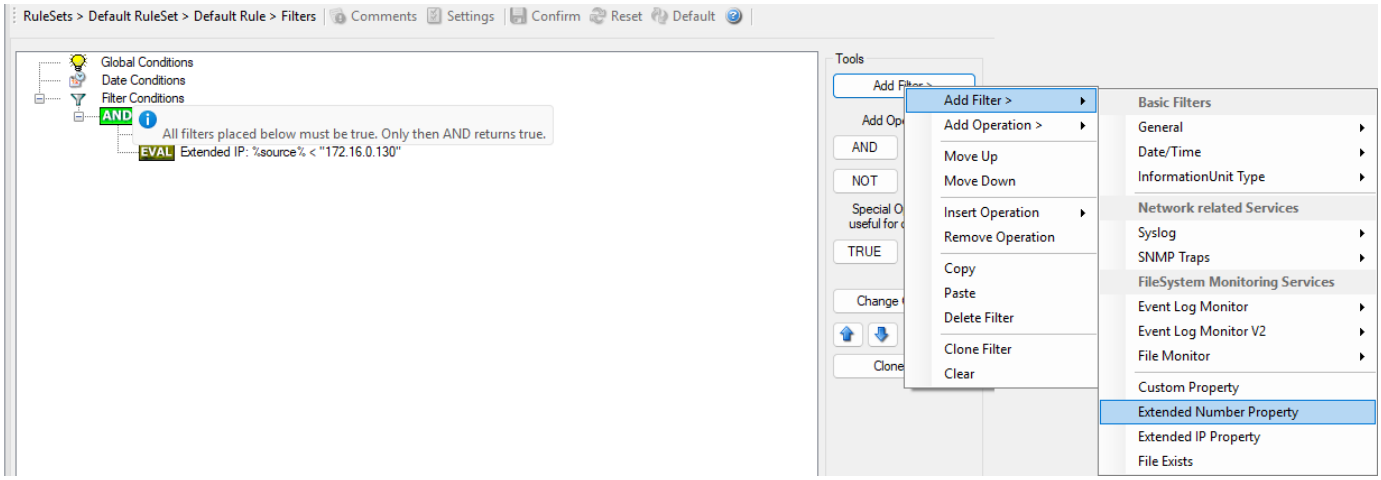
**Custom Property**

As the name suggests it is a “Custom Property”. Internally in WinSyslog all values are stored in properties. For example the main message is stored in a property called “msg”. By using this dialog you can access properties which are dynamic (Like those from SNMP Trap Monitor when using V2 protocol).

This filter is of type string.

## Extended Number Property

Extended Number Property specific filter is described here.



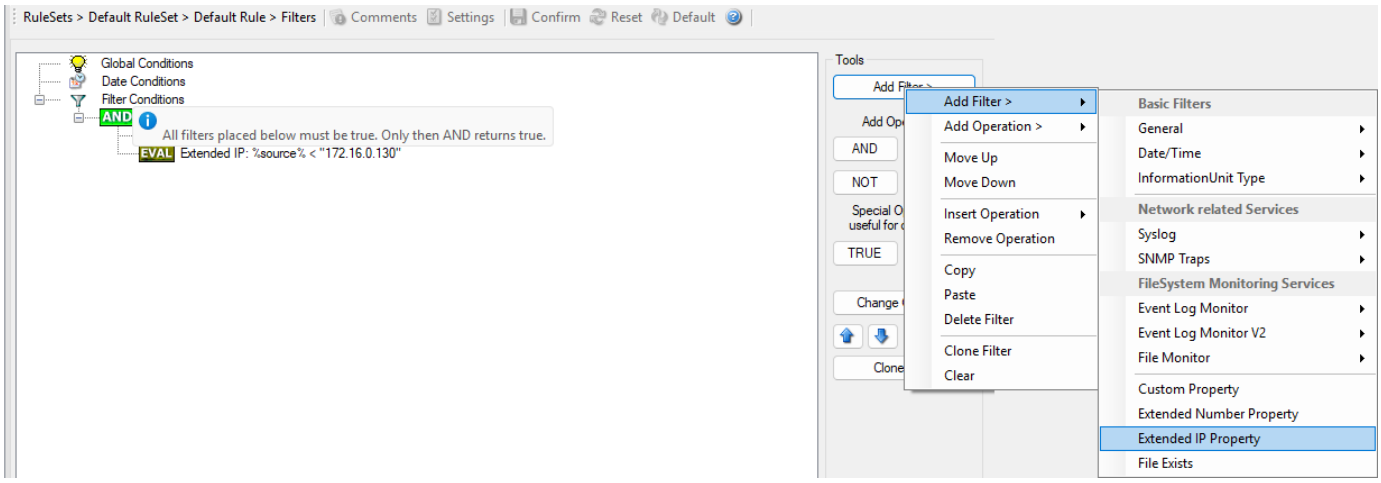

*WinSyslog Filter Conditions - Extended Number Property*

**Extended Number Property**

As the name suggests it is a “Extended Number Property”. Internally in WinSyslog all values are stored in properties. For example the main message is stored in a property called “msg”. By using this dialog you can access properties which are dynamic (Like those from SNMP Trap Monitor when using V2 protocol).

This filter is of type numeric.

## Extended IP Property




*WinSyslog Filter Conditions - Extended IP Property*

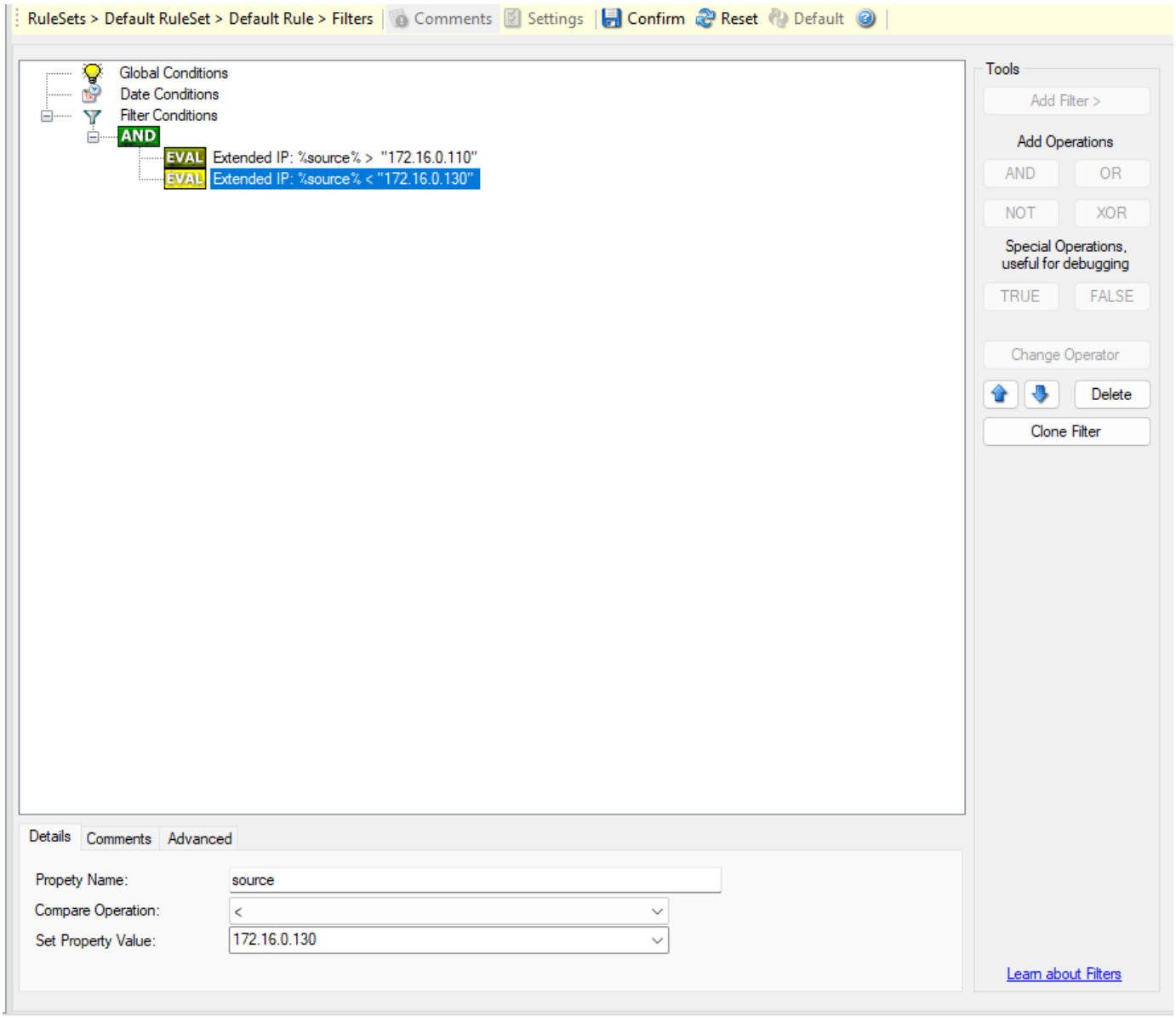
### Extended IP Property filter settings

The IP Filter can basically work on any property, but we recommend to only use it on the %source% property, as we usually can be sure that this contains a valid IP Address or hostname. The IP Filter can filter against hostnames and IP Addresses, hostnames are automatically resolved using the internal DNSCache (for obvious performance reasons). If you are going to use a different or custom property, please make sure, that the data in the property is a valid IP Address.

Available compare operations for the IP Filter Type are:

Equal (=): The IP Address must match the one you configured in the Property Value field. Not Equal (!=): The IP Address must not match the one you configured in the Property Value field. Higher (>): The IP Address must be higher than the one you configured in the Property Value field. You can use IP Address Formats like: `192.168.0.10, 192.168.0, 192.168 or even 192`. It depends on what IP Ranges you are going to filter for. Lower (<): The IP Address must be lower than the one you configured in the Property Value field. You can use IP Address Formats like: `192.168.0.10, 192.168.0, 192.168 or even 192`. It depends on what IP Ranges you are going to filter for.

If you want to filter for IP Ranges, I recommend to use two filters to define the range, one filter with the “Higher (>)” compare operation and one with the “Lower (<)” compare operation. This could look like the following:

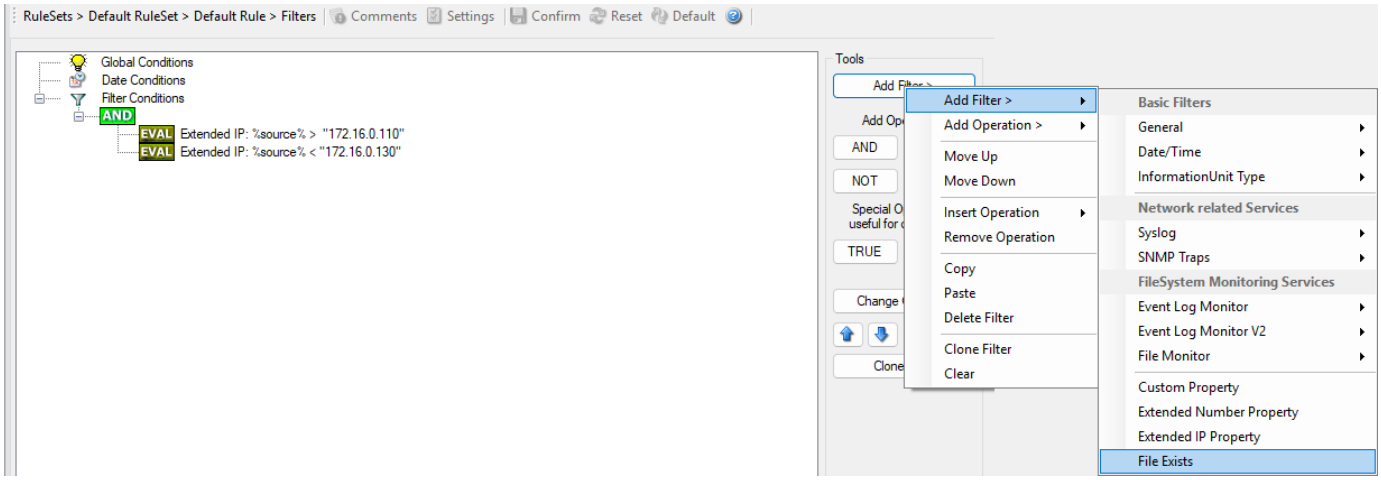



*WinSyslog Filter Conditions - Filtering for an IP Range*

The filter you can see here will accept all IPs which lie between `172.16.0.110` AND `172.16.0.130`. That means, that for every IP that matches these two conditions, the whole filter will evaluate to true and therefore the message will be processed. If the filter does not evaluate to true, the rule will be aborted and the message is sent to the next rule.

## File Exists

Filter setting by string.



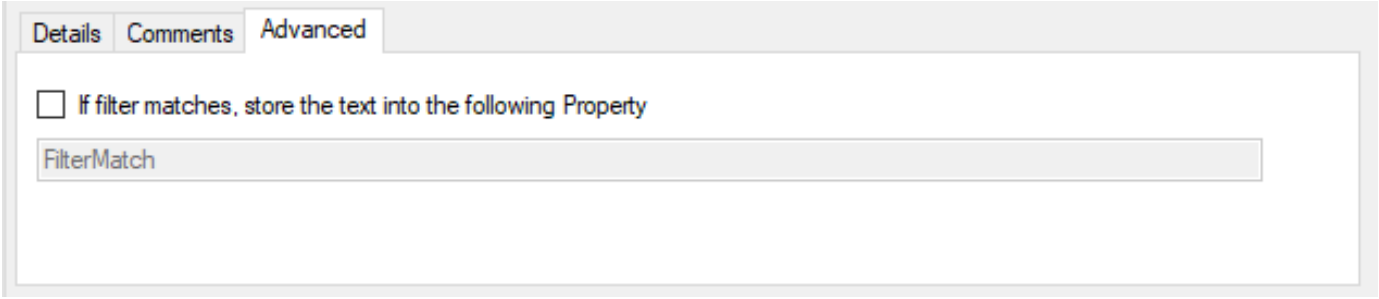

*WinSyslog Filter Conditions - File Exists*

**File Exists**

With this Filter you can simply check if a file exists or not. You can directly enter the file and its location or you can use the browse-button to find it.

### Store Filter Results

How to store Filter Results is described here.




- Filter Conditions - Store Filter Results*

**Store Filter Results**

If a filter matches, you can now store the result of the match into a custom property.

This custom property can be used in Actions later.

## Actions

Actions define what WinSyslog does after a rule matches an event. They are the output and follow-up stage of processing.

A single rule can contain multiple actions. Actions run in listed order, so the sequence can matter when one action changes data or affects later behavior.

Use the sections below to choose the right action type for storage, forwarding, notification, or internal processing.

### Storage actions

## ODBC Database Options

Use this action to write matched events or messages to a database through ODBC.

The ODBC database action is an integration feature. It can write to the built-in Adiscon default schema or to a user-defined schema in any supported ODBC database. Use the default schema when you want the fastest supported setup or compatibility with Adiscon tools. Use custom mapping when the product needs to write into an existing database design.

### Common usage patterns

- **Default schema:** Use the built-in `SystemEvents` and `SystemEventProperties` tables when you want the shortest supported setup or when downstream Adiscon tools expect the standard schema.
- **Custom schema integration:** Map event properties to an existing table with your own column names and data types.
- **Microsoft SQL Server stored procedures:** Use the call-statement option only when your SQL Server design requires a stored procedure instead of a standard `INSERT` statement.

### Out of scope

This action does not design your database for you. It does not decide table layout, indexes, retention policy, reporting logic, or broader analytics architecture. For custom integration, you own the destination schema and the mapping decisions.

### Before you start

- Install a supported ODBC driver on the Windows host that runs the service.
- Create an **ODBC System DSN** for the target database. User DSNs and file DSNs are not suitable for the service path.
- Verify that the database server is reachable and that the configured credentials have the required permissions.
- Decide whether the action should:
  - create and use the default Adiscon tables, or
  - write into an existing user-defined table

### Minimal action path

1. Create and test an ODBC **System DSN** outside the product.
2. Add a **Write to Database** action to the relevant ruleset.
3. Configure the DSN, credentials, and connection settings.
4. Choose one of these paths:
   - keep the default schema and use **Create Database**, or
   - set the table name and field list for a custom schema
5. Save and apply the configuration.
6. Send a matching test event or message and verify that rows are inserted.

### Default schema versus custom schema

#### Default schema

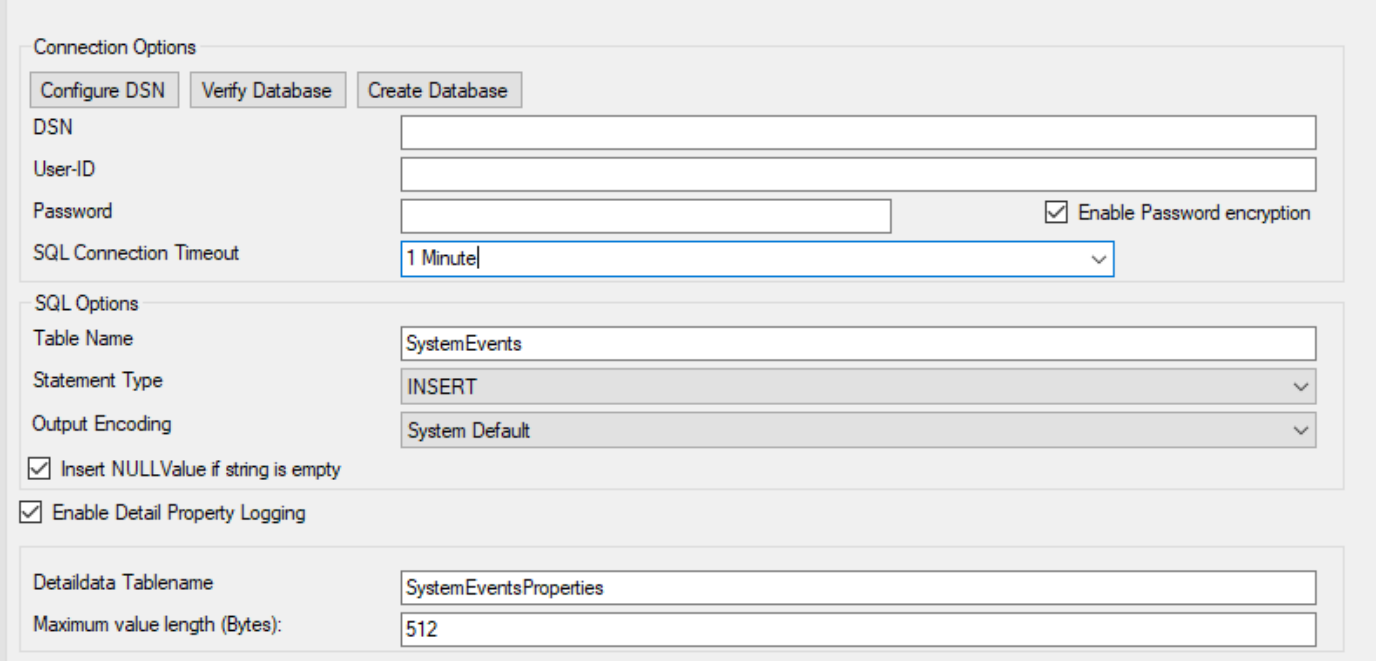
Use the default schema when you want a predictable starting point or when you need compatibility with other Adiscon components that expect the standard table layout. In this path, the action can create the default tables for you.

#### Custom schema integration

Use a custom schema when your organization already has a database design that WinSyslog, EventReporter, or WinSyslog must write into. In this path, the action does not infer the schema. You must set the target table name and map each field deliberately.

If you diverge from the default schema, do not assume that Adiscon tools that expect the standard layout will continue to work unchanged.

### Connection options




*Action - ODBC Database Connection*

### Buttons

**Configure DSN**

Opens the Windows ODBC administrator so you can add, edit, or remove data sources.

**Verify Database**

Attempts to connect to the configured ODBC System DSN with the current settings. Use this before you save the action into production.

**Create Database**

Creates the default Adiscon tables in the target database. Use this only when you intentionally want the default schema.

### DSN

**File Configuration field:**

szODBCDsn

**Description:**

Name of the ODBC **System DSN** used for the database connection. The DSN must already contain the correct driver and target-database connection details.

### User-ID

**File Configuration field:**

szODBCUid

**Description:**

User name for database authentication, if the DSN and driver require it.

### Password

**File Configuration field:**

szODBCPwd

**Description:**

Password for the configured user ID. Use an account with only the permissions needed for this action.

### Enable Encryption

**File Configuration field:**

nODBCEnCryption

**Description:**

Stores the configured ODBC password encrypted instead of plaintext. Enable this unless you have a documented reason not to.

### SQL Connection Timeout

**File Configuration field:**

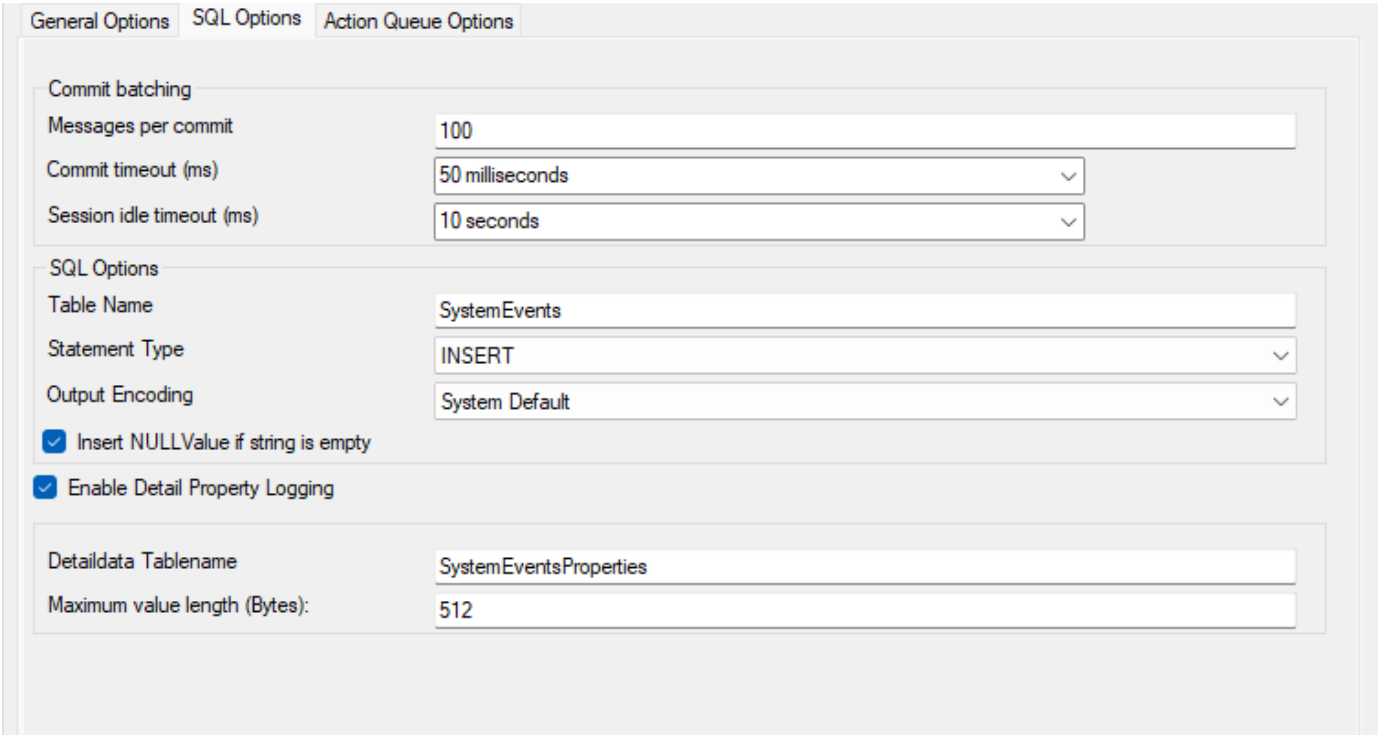
nSQLConnectionTimeOut

**Description:**

Maximum time to wait while opening the database connection.

### SQL options

The client organizes ODBC database settings across **General Options**, **SQL Options**, and **Action Queue Options** tabs.




*Action - ODBC Database SQL Options*

### Commit batching (SQL Options tab)

**File Configuration field:**

nCommitBatchSize

**Description:**

Number of messages to batch before committing (Messages per commit). Increase for higher throughput when the database can sustain larger batches; decrease for more frequent commits.

**File Configuration field:**

nCommitTimeoutMs

**Description:**

Maximum time in milliseconds to wait before flushing a partial batch.

**File Configuration field:**

nSessionTimeoutMs

**Description:**

Idle timeout in milliseconds for the database session.

### Table Name

**File Configuration field:**

szTableName

**Description:**

Target table name for database writes. Keep the default `SystemEvents` when you use the built-in schema. Set it to your existing table name when integrating with a custom schema.

### SQL Statement Type

**File Configuration field:**

nSQLStatementType

**Description:**

Selects whether the action uses a normal `INSERT` statement or a Microsoft SQL Server call statement for stored procedures. The call-statement path is Microsoft SQL Server specific.

### Output Encoding

**File Configuration field:**

nOutputEncoding

**Description:**

Controls how string data is encoded when written. In most environments, **System Default** is the correct setting unless you have a confirmed character-set requirement.

### Insert NULL Value if string is empty

**Description:**

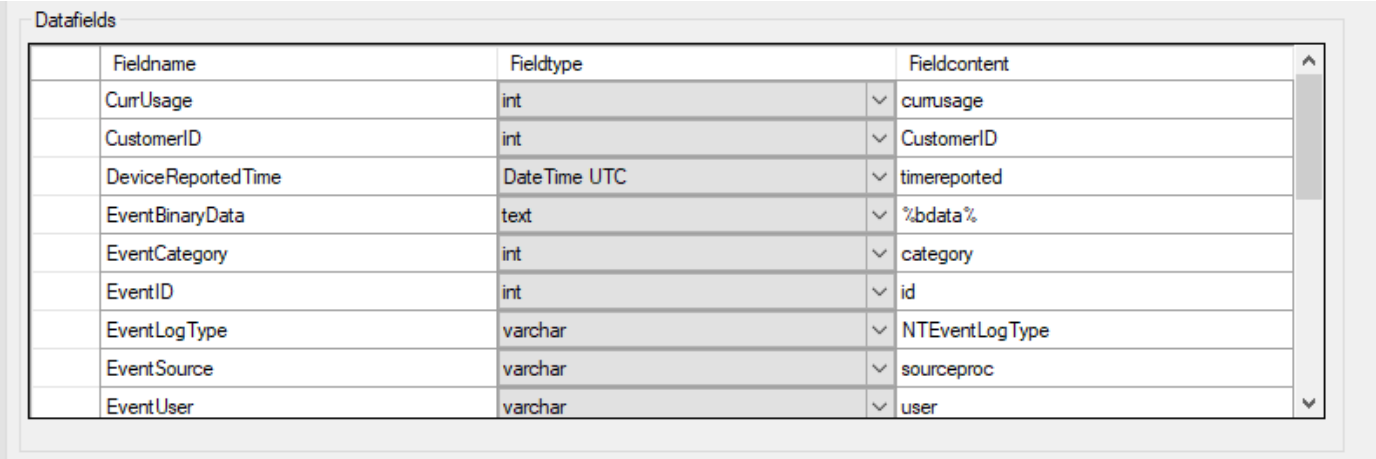
Writes `NULL` instead of an empty string for empty string properties. Use this only if your schema and downstream queries intentionally distinguish between empty text and `NULL`.

### Datafields

The field list controls how event properties are written into the destination table. This is the most important part of custom integration work.

For the default schema, the built-in field list already reflects the standard Adiscon table layout. For a custom schema, keep only the rows that correspond to actual destination columns and adjust them deliberately.

For string data types, you can use the property replacer. For example, the expression `%msg:1:200%` stores only the first 200 characters of the message. For simple mappings, use the relevant event property directly.




*Action - ODBC Database Datafields*

### Fieldname

**File Configuration field:**

szFieldName_[n]

**Description:**

Database column name in the destination table.

### Fieldtype

**File Configuration field:**

nFieldType_[n]

- 1 = varchar
- 2 = int
- 3 = text
- 4 = DateTime

**Description:**

Data type of the destination column. It must match both the database schema and the kind of property you are storing.

### Fieldcontent

**File Configuration field:**

szFieldContent_[n]

**Description:**

Event property or property-replacer expression written into the destination column. See event properties and property access and replacer syntax.

### Practical mapping guidance

For a custom syslog-oriented table, a minimal mapping often includes:

- a timestamp column populated from `timegenerated` or `timereported`
- a source column populated from `source`
- a severity column populated from `syslogpriority`
- a tag or application column populated from `syslogtag` or `syslogappname`
- a message column populated from `msg`

If a destination column is shorter than the source property, truncate or transform the value explicitly instead of hoping the driver or database will do the right thing.

### Detail property logging

### Enable Detail Property Logging

**File Configuration field:**

nPropertiesTable

**Description:**

Writes non-standard properties into a separate detail table. This can be useful when additional event metadata must be retained, but it also increases write volume.

### Detaildata Tablename

**File Configuration field:**

szPropertiesTableName

**Description:**

Table name used for detail-property logging. In the default schema, this is typically `SystemEventProperties`.

### Maximum value length (Bytes)

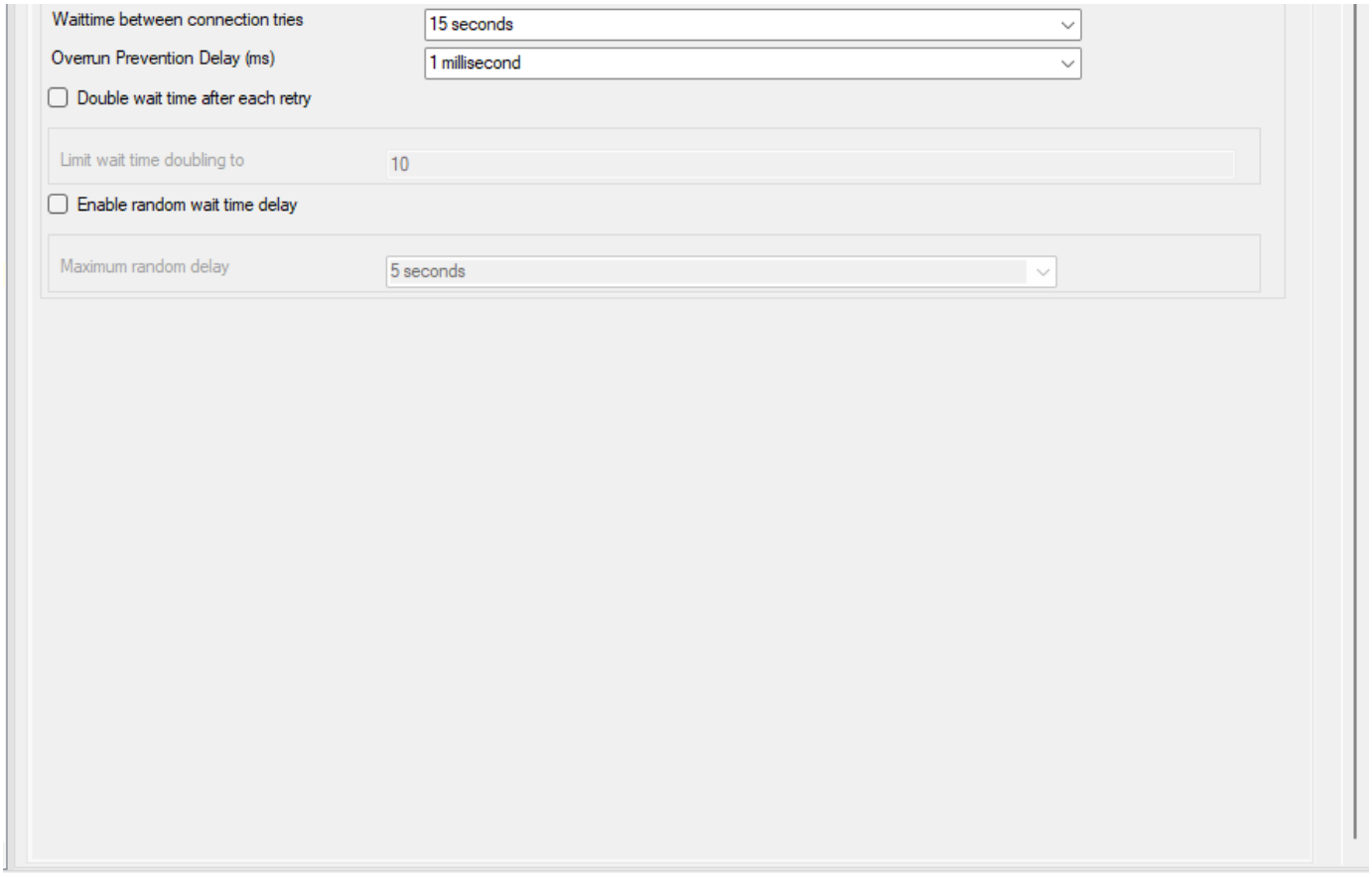
**File Configuration field:**

nMaxValueLength

**Description:**

Maximum size in bytes for values written into the detail-property table.

### Action Queue Options




*WinSyslog - ODBC database Action Queue Options*

### Use Diskqueue if connection to database fails

**File Configuration field:**

nUseDiscQueue

**Description:**

Stores pending writes on disk when the database path is temporarily unavailable.

### Split files if this size is reached

**File Configuration field:**

nDiskQueueMaxFileSize

**Description:**

Files are split when they reach this size in bytes. The default is 10485760 (10 MB). The maximum per file is 2147483648 bytes (2 GB). The total diskqueue size is limited by free disk space.

### Diskqueue Directory

**File Configuration field:**

szDiskQueueDirectory

**Description:**

Directory used to store queue files for pending database writes.

### Waittime between connection tries

**File Configuration fields:**

nDiskCacheWait

**Description:**

Minimum wait time before the action retries the database connection after a failure.

### Overrun Prevention Delay (ms)

**File Configuration field:**

nPreventOverrunDelay

**Description:**

Optional delay between replayed queue writes to avoid overwhelming the target database after recovery.

### Double wait time after each retry

**File Configuration field:**

bCacheWaittimeDoubling

**Description:**

Doubles the retry wait time after each failure.

### Limit wait time doubling to

**File Configuration field:**

nCacheWaittimeDoublingTimes

**Description:**

How many times the wait time is doubled after a failed connection try. This is a count, not seconds. The range is 1 to 100, and the default is 10.

### Enable random wait time delay

**File Configuration field:**

bCacheRandomDelay

**Description:**

Adds a randomized delay to retry timing. This can reduce synchronized retry spikes when many senders reconnect at the same time.

### Maximum random delay

**File Configuration field:**

nCacheRandomDelayTime

**Description:**

Upper bound for the additional randomized retry delay.

### Verification

- Use **Verify Database** before enabling production traffic.
- Send a matching test event or message after saving the action.
- Query the destination table and confirm that:
  - rows are inserted
  - values appear in the expected columns
  - data types and lengths are compatible with the schema

### Common pitfalls

- Using a user DSN instead of a **System DSN**
- Leaving the default field list unchanged while targeting a custom table
- Using **Create Database** when the goal is an existing custom schema
- Mapping text properties into integer or datetime columns
- Using the SQL Server call-statement mode on non-Microsoft SQL Server targets
- Forgetting that custom schemas may break compatibility with tools that expect the default Adiscon layout

## OLEDB Database Action

Use this action to write matched events or messages to a database through an OLEDB provider.

This action serves the same main use cases as ODBC Database Options, but it connects through OLEDB instead of an ODBC System DSN. It can write to the built-in Adiscon default schema or to a user-defined schema. Provider availability depends on your Windows environment and the database vendor's current OLEDB support.

### When to choose OLEDB

- You already have a supported OLEDB provider for the target database.
- Your environment standardizes on OLEDB rather than ODBC.
- You need the same database-writing and field-mapping behavior but through an OLEDB connection path.

Use the ODBC action instead when your preferred or only supported driver path is ODBC.

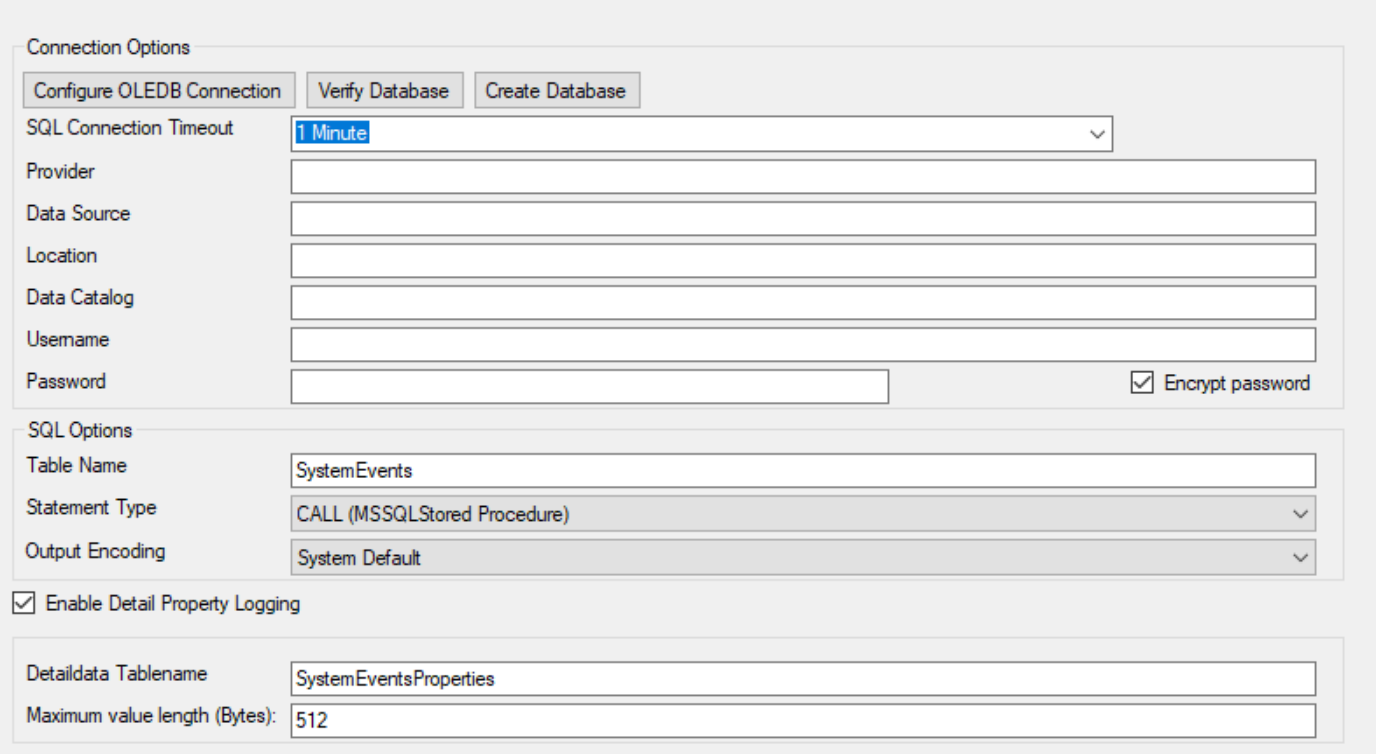
### Before you start

- Verify that the required OLEDB provider is installed on the Windows host.
- Confirm the server, database, and authentication details that the provider expects.
- Decide whether you want the default Adiscon schema or an existing custom schema.
- Ensure the target account has the required database permissions.

### Minimal action path

1. Configure the OLEDB connection.
2. Use **Verify Database** to test the connection.
3. Choose one of these paths:
    - use **Create Database** for the default schema, or
    - set the table name and field list for a custom schema
4. Save and apply the configuration.
5. Send a matching test event or message and verify that rows are inserted.

### Connection options




*Action - OLEDB Database Connection*

### Buttons

**Configure OLEDB Connection**

Starts the OLEDB configuration wizard for the provider and connection string.

**Verify Database**

Tests the current OLEDB connection settings.

**Create Database**

Creates the default Adiscon tables in the target database. Use this only when you intentionally want the default schema.

### SQL Connection Timeout

**File Configuration field:**

nSQLConnectionTimeOut

**Description:**

Maximum time to wait while opening the database connection.

### Provider

**File Configuration field:**

szProvider

**Description:**

OLEDB provider name. Use a provider that is actually installed and supported in your environment.

### Data Source

**File Configuration field:**

szDataSource

**Description:**

Server, instance, or provider-specific data source identifier.

### Location

**File Configuration field:**

szLocation

**Description:**

Optional OLEDB location setting if your provider requires it.

### Data Catalog

**File Configuration field:**

szDataCatalog

**Description:**

Database name or catalog, depending on the provider.

### Username

**File Configuration field:**

szUsername

**Description:**

User name for database authentication, if required by the provider.

### Password

**File Configuration field:**

szPassword

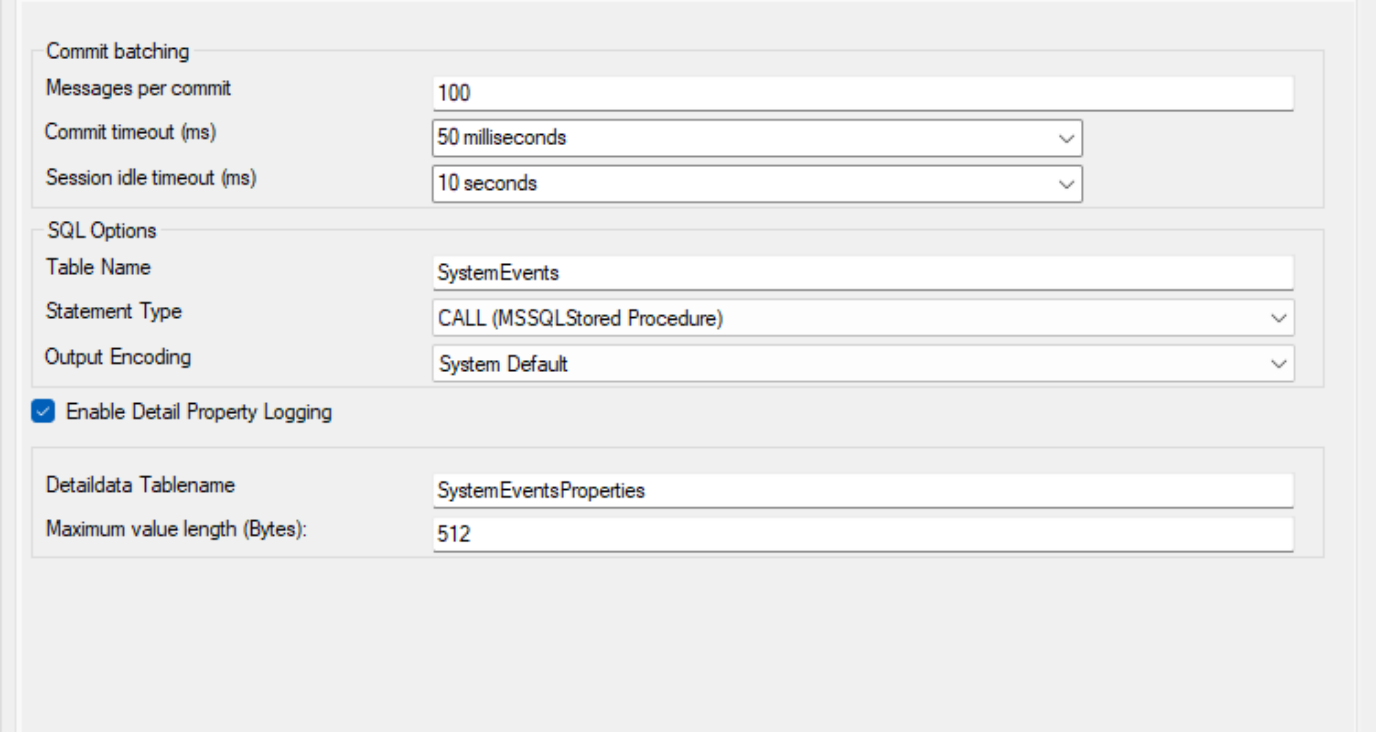
**Description:**

Password for the configured user.

### Encrypt password

**Description:**

Enable password encryption if your build exposes this option. As with ODBC, prefer encrypted storage unless you have a documented reason not to.




*Action - OLEDB Database SQL Options*

### Commit batching (SQL Options tab)

**File Configuration field:**

nCommitBatchSize

**Description:**

Number of messages to batch before committing (Messages per commit). Tune for throughput versus commit frequency under load.

**File Configuration field:**

nCommitTimeoutMs

**Description:**

Maximum time in milliseconds to wait before flushing a partial batch (Commit timeout).

**File Configuration field:**

nSessionTimeoutMs

**Description:**

Idle timeout in milliseconds for the database session (Session idle timeout).

### Table Name

**File Configuration field:**

szTableName

**Description:**

Target table name for database writes. Keep the default `SystemEvents` when you use the built-in schema. Set it to your existing table when integrating with a custom schema.

### Statement Type

**File Configuration field:**

nSQLStatementType

**Description:**

Selects whether the action uses a standard `INSERT` statement or a Microsoft SQL Server call statement for stored procedures. The call-statement path is Microsoft SQL Server specific.

### Output Encoding

**File Configuration field:**

nOutputEncoding

**Description:**

Controls how string data is encoded when written. In most environments, **System Default** is the correct setting unless you have a confirmed character-set requirement.

### Data mapping and custom schemas

The field list works the same way as in ODBC Database Options. It controls which event properties are written to which destination columns.

For custom integration:

- set the table name to your existing table
- keep only the fields that exist in that table
- make each field name, field type, and field content match the destination schema deliberately

For string fields, you can use property-replacer expressions such as `%msg:1:200%` when you need truncation or transformation.

If you use the default schema, keep the default field list unchanged unless you understand the compatibility impact on tools that expect the standard Adiscon layout.

### Detail property logging

**File Configuration field:**

nPropertiesTable

**Description:**

Writes non-standard properties into a separate detail table. This increases write volume and is usually needed only when you intentionally want those additional properties retained.

### Detaildata Tablename

**File Configuration field:**

szPropertiesTableName

**Description:**

Table name used for detail-property logging. In the default schema, this is typically `SystemEventProperties`.

### Maximum value length (Bytes)

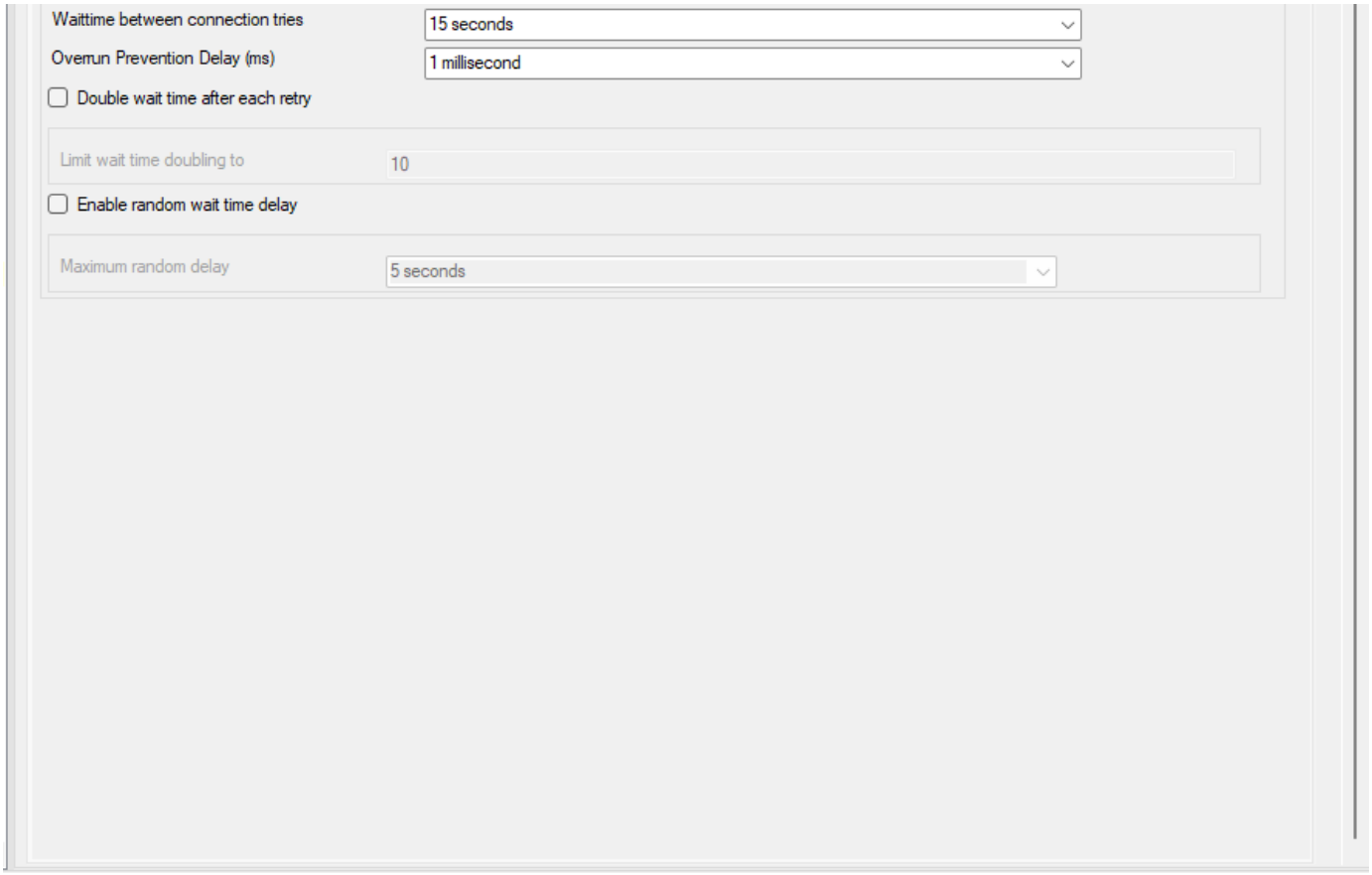
**File Configuration field:**

nMaxValueLength

**Description:**

Maximum size in bytes for values written into the detail-property table.

### Action Queue Options




*WinSyslog - OLEDB database Action Queue Options*

### Use Diskqueue if connection to database fails

**File Configuration field:**

nUseDiscQueue

**Description:**

Stores pending writes on disk when the database path is temporarily unavailable.

### Split files if this size is reached

**File Configuration field:**

nDiskQueueMaxFileSize

**Description:**

Files are split when they reach this size in bytes. The default is 10485760 (10 MB). The maximum per file is 2147483648 bytes (2 GB). The total diskqueue size is limited by free disk space.

### Diskqueue Directory

**File Configuration field:**

szDiskQueueDirectory

**Description:**

Directory used to store queue files for pending database writes.

### Waittime between connection tries

**File Configuration fields:**

nDiskCacheWait

**Description:**

Minimum wait time before the action retries the database connection after a failure.

### Overrun Prevention Delay (ms)

**File Configuration field:**

nPreventOverrunDelay

**Description:**

Optional delay between replayed queue writes to avoid overwhelming the target database after recovery.

### Double wait time after each retry

**File Configuration field:**

bCacheWaittimeDoubling

**Description:**

Doubles the retry wait time after each failure.

### Limit wait time doubling to

**File Configuration field:**

nCacheWaittimeDoublingTimes

**Description:**

How many times the wait time is doubled after a failed connection try. This is a count, not seconds. The range is 1 to 100, and the default is 10.

### Enable random wait time delay

**File Configuration field:**

bCacheRandomDelay

**Description:**

Adds a randomized delay to retry timing. This can reduce synchronized retry spikes when many senders reconnect at the same time.

### Maximum random delay

**File Configuration field:**

nCacheRandomDelayTime

**Description:**

Upper bound for the additional randomized retry delay.

### Common pitfalls

- Assuming OLEDB is required when a supported ODBC path is simpler
- Relying on provider names or examples from older Windows environments without verifying that the provider is still installed and supported
- Using the default field list unchanged while targeting a custom table
- Expecting the action to design a custom schema automatically

### File Logging Options

This configuration dialog is available both in the defaults section as well as with file logging actions.

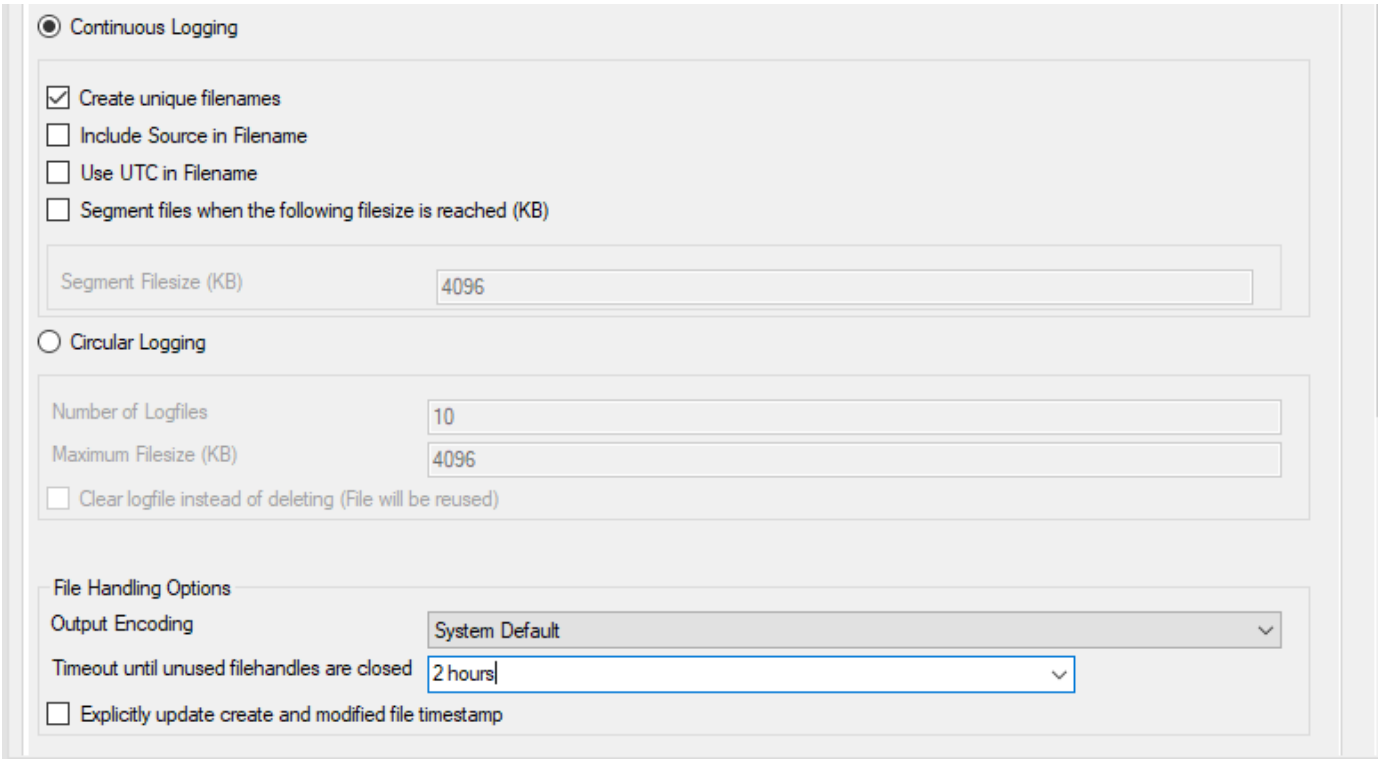
File logging is used to write text files of received messages. One file per day is written. New entries are appended to the end of the file.

File locks are released when currently no data is written. Therefore, other applications can access the files while the service is running. However, please be sure that the other applications do not place a file-lock onto it. Popular WordPad does so. In this case, the service will not be able to log any further messages (an error event is written to the Windows Event Log in this case). We recommend copying the file when accessing it at runtime - or use notepad.exe, which does not place file-locks on the files it opens.

**The filename is build as follows:**

`<FilePathName><FileBaseName>-year-month-day.<FileExtension>`

Parameters in the brackets can be configured via dialog shown below:




*WinSyslog - File Logging filename options*

### Enable Property replacements in Filename

**File Configuration field:**

nEnablePropertyFileName

**Description:**

By activating this option, you can use properties within the file or pathname like `%source%` and all the others. For example: File Path Name can be `F:\syslogs\%source%` File Base Name can be `IIS-%source%`

If your source is `10.0.0.1`, that writes the following file: `F:\syslogs\10.0.0.1\IIS-10.0.0.1.log`

The path `f:\syslogs\10.0.0.1` was generated because the source property was used inside the path.

Please Note that you can use ANY property inside the path and base name. event properties are described in the property replacer section.

### File Path Name

**File Configuration field:**

szFilePath

**Description:**

The base path (directory) of the file. Please see above for exact placement. Default is `c:\temp`. The Insert Menu entry allows you to create “Dynamic Directories”. For example:

File Path Name can be **``F:syslogs%source%``**

event properties are described in the property replacer section.

**On network paths:** The File Logging action can also work on network storages. There are two ways of storing log files in a network path.

1. Direct the action to a full UNC path. In this case, make sure the system account with which the service is running is able to access the network path or the service will fail to access with a permission error. Sample path: `\Hostname\folder1\folder2\`
2. Map the UNC path to a local drive letter in Windows. In this case, the path will look like a regular local path, but actually points to a network location. This requires a workaround, which is to run a scheduled task at system startup under Local System and perform a net use specifying the user and password of the share. Else, the service will not be able to access the mapped UNC path, because the mapping usually happens for interactive sessions only.

### File Base Name

**File Configuration field:**

szFileBaseName

**Description:**

The base name of the file. Please see above for exact placement. Default is “MonitorWare”. The Insert Menu entry allows you to recreate “Dynamic Base Filenames”. For example:

File Base Name can be `IIS-%source%`

### File Extension

**File Configuration fields:**

szFileExtension

**Description:**

The extension to be used when writing the file. Please see above for exact placement. Default is `.log`.

### Continuous Logging

**Description**

When enabled log files will not be overwritten, there is a single file with consistent file name. See below checkboxes to choose in which cases a new file should be created.

### Create unique Filenames

**File Configuration field:**

nUniqueFileName

**Description:**

If checked, a unique file name is created for each day. This is done by adding the current date to the base name.

If left unchecked, the date is not added and as such, there is a single file with consistent file name. Some customers that have custom scripts to look at the file name use this.

### Include Source in Filename

**File Configuration field:**

nIncludeSourceInFilename

**Description:**

This works together with the “Create unique Filenames” setting. If checked, the file name generation explained above is modified. The source of the Syslog message is automatically added to the file name.

This feature has been introduced because many customers would like to have separate log files for each device. While this can be achieved with multiple rules, it is much more straight forward with this single checkbox. If it is checked, the messages are automatically written to separate files and the file name includes the originating device information.

### Use UTC in Filename

**File Configuration field:**

nUseUTCInFileName

**Description:**

This works together with the “Create unique Filenames” setting. If unique names are to be created then select the “Use UTC in Filename” option, in this case the file name is generated on the basis of universal coordinated time (UTC) or on local time. UTC was formerly referred to as “GMT” and is the basis of the time zone system. For example, New York, USA is 5 hours behind UTC. Therefore, if it is 12 noon in New York, the UTC time is 5pm.

When it comes to log file creation, it means that the date is computed on UTC. Taking the same example, if the “Use UTC in Filename” is checked, the log file name would roll over to the next date at 7 pm New York time. If it were unchecked, the rollover would occur exactly at midnight New York time (5 am UTC).

Using UTC for file name creation can be helpful if log files are written among different time zones and later consolidated. Using UTC ensures a consistent time notation across all log files.

**Please note that this setting does affect the file name creation only. A different setting controls the dates recorded inside the file.**

### Segment files when the following file size is reached (KB)

**File Configuration field:**

nSegmentFileEnable

**Description:**

Files are segmented if the defined file size: Segment Filesize (KB) is reached. A sequence number is appended to the file name: _1 to _n.

### Circular Logging

**File Configuration field:**

nCircularLogging

**Description:**

If enabled, log files are created and overwritten in a cycle.

### Number of Log Files

**File Configuration field:**

nNumberOfLogfiles

**Description:**

Once the last log file is reached, circular logging begins and overwrites the first log file again. If set to 0, log files will not be rotated but can still be processed by Rotate Post Processing (for example compression or backup) along with the Rotate Conditions.

### Maximum Filesize (KB)

**File Configuration field:**

nMaxFileSize

**Description:**

Max filesize of a log file, once this size is reached a new logfile is created.

### Clear logfile instead of deleting (File will be reused)

**File Configuration field:**

nReUseFile

**Description:**

This option causes the File Action to truncate the log file instead of deleting and recreating it.

### File Handling Options

### Output Encoding

**File Configuration field:**

nOutputEncoding

**Description:**

This setting is most important for Asian languages. A good rule is to leave it at “System Default” unless you definitely know you need a separate encoding. “System Default” works perfectly in the far majority of cases, even on Asian (e.g. Japanese) Windows versions.

### Timeout until unused filehandles are closed

**File Configuration field:**

nCleanFileHandlesTimeout

**Description:**

When dynamic filenames are used, filehandles are cached internally to avoid massive amount of File open/close operations. This timeout specifies after which time handles should be finally closed if not used anymore. Each write to a file will reset the timeout counter for the current filehandle.

### Explicitly update create and modified file Timestamp

**File Configuration field:**

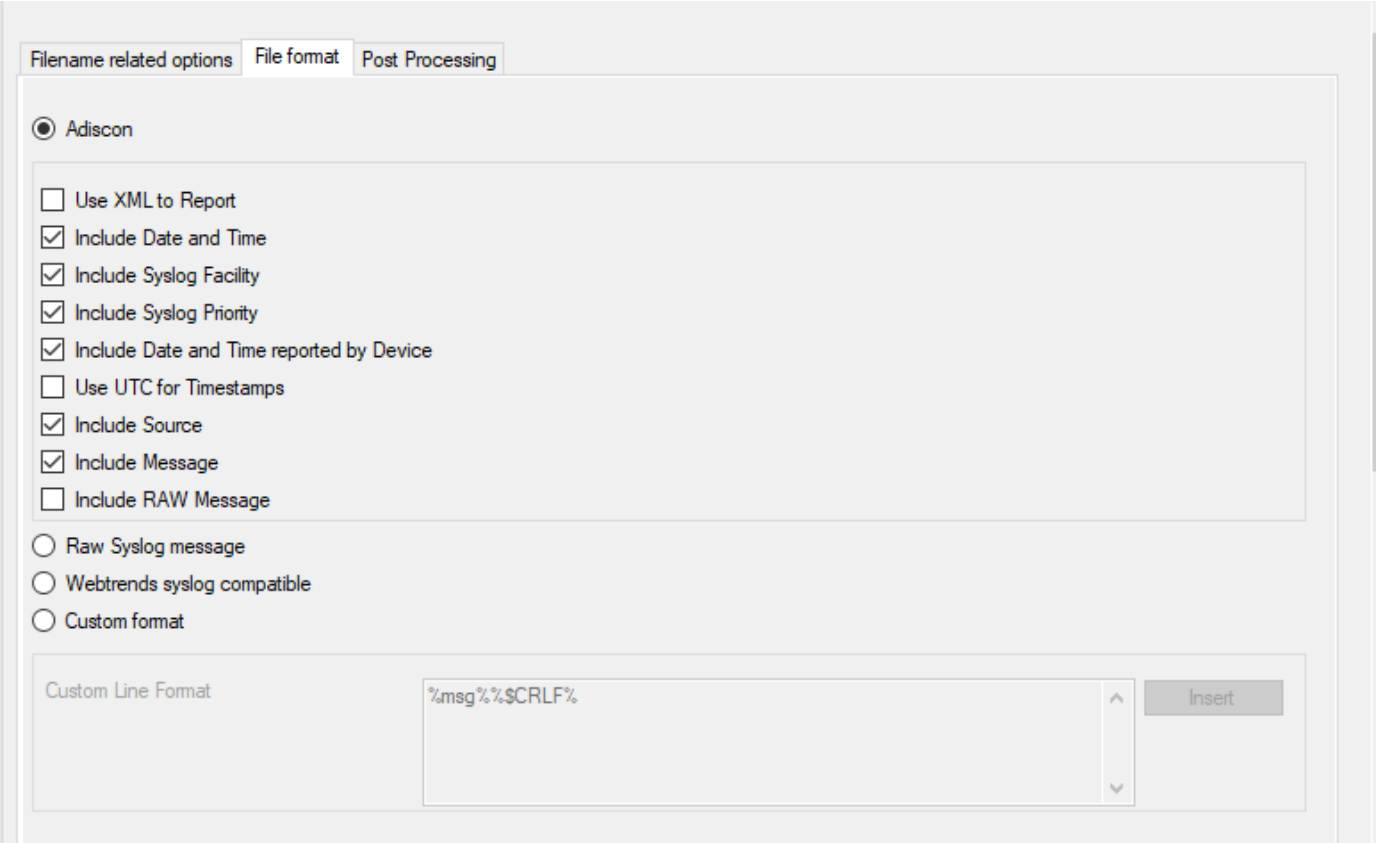
nEnableUpdateFileTime

**Description:**

If the checkbox is not selected the operating system updates the timestamps for creating and modifying files. In cases where the filesystem does not do this reliably, the checkbox can be selected. Now the service itself updates the timestamps for creating and modifying files.

### File Format

The format in which the log file is written can be selected here. The default is “Adiscon”, which offers most options. Other formats are available to increase log file compatibility to third party applications.




- Action - File Logging File Format*

### Adiscon

> **Note**
>
> Any other format besides “Adiscon Default” are fixed formats. As such, if it is selected, all other formatting options do not apply and consequently are turned off.

**The following options are possible:**

#### Use XML to Report

**File Configuration field:**

nUseXMLtoReport

**Description:**

If checked, the message part includes a complete XML-formatted information record. It includes additional information like timestamps, syslog facility and priority, and others in an easy to parse format. If XML output format is selected, you might consider turning all other information fields off, as they are already included in the XML stream. However, this is not a requirement.

#### Use UTC for Timestamps

**File Configuration field:**

nUseUTCForTimestamps

**Description:**

Please see the definition of utc above at “Use UTC in Filename”. This setting is very similar. If checked, all time stamps are written in UTC. If unchecked, local time is used instead. Again, UTC is useful if logs written in multiple time zones are to be consolidated.

### Include <Fieldname>

**File Configuration field:**

- nFileDateTime
- nFileFacility
- nFilePriority
- nFileDateTimeReported
- nFileSource
- nIncludeMessage
- nIncludeRAWMessage

**Description:**

The various “include” settings controls are used to specify the fields which are to be written to the log file. All fields except the message part itself are optional. If a field is checked, it is written to the log file. If unchecked, it will not be written. All fields are comma-delimited.

Please note the difference between the “Date and Time” and “Date and Time reported by Device”. Both are timestamps. Either both are written in local time or utc based on the “Use UTC for Timestamps” check box. However, “Date and Time” is the time when the product received the message. Therefore, it is always a consistent value.

In contrast, the “Date and Time Reported by Device” is a timestamp taken from the actual message. As such, it is dependent on the reporting device clock, which might be off. In addition, in the case of Syslog messages, there is no time zone information within the device reported timestamp. As such, if devices from multiple time zones are reporting, the timestamp information is not consistent. This is due to Syslog design as of rfc 3164. The Syslog server can be configured to ignore the RFC in this case and provide a consistent time stamp. However, from the view of the log file writer, the “Date and Time Reported by Device” might not be as trustworthy as the “Date and Time” field. Nevertheless, it might also be more useful than the former one. This is the reason both timestamps are present and can individually be selected.

The “Include Message” and “Include RAW Message” fields allow customizing the message part that is being written. The raw message is the message as – totally unmodified, was received. This might be useful if a third party application is expecting raw Syslog entries. The message itself is just that part of the Syslog message that is being parsed as message. That is without e.g. host information or a tag value. Please note that we recommend selecting only one of these options, as otherwise two message fields are written. Similarly, if none is selected no message is written at all. Please note that we support these configurations, too – there might be a legitimate need for them.

### Raw Syslog message

The “Raw Syslog message” format writes raw Syslog format to the log file. That is, each line contains the Syslog message as of RFC 3164. No specific field processing or information adding is done. Some third party applications require that format.

### Webtrends syslog compatible

The “WebTrends Syslog compatible” mimics the format that WebTrends applications expect. Please note that we only mimic the log file format. It is still the job of the reporting device (most notable firewall) to generate the correct WebTrends WELF format. The “WebTrends” format is supported because many customers would like to use product enhanced features while still having the ability to work with WebTrends.

### Custom format

The “Custom format” allows you to customize formats to increase log file compatibility for third party applications. When you choose this option then Custom line format is enabled.

### Custom Line Format

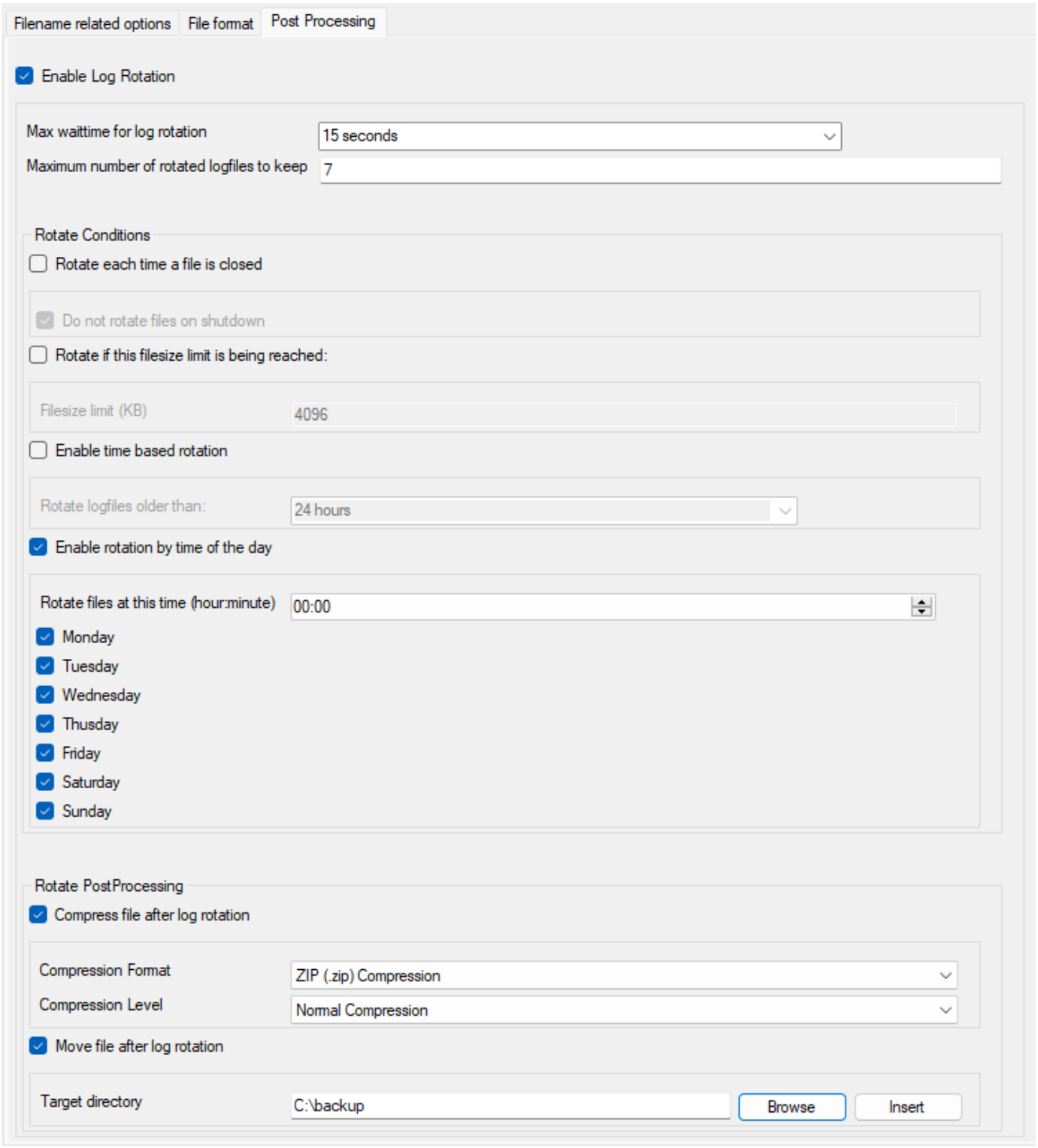
**File Configuration field:**

szLineFormat

**Description:**

Custom Line Format enables you to fully customize the output for the log file. The Insert Menu entry provides further options and they only work in custom line format. Default value is `%msg%%$CRLF%`.

### Post Processing




- Action - File Logging Post Processing*

### Enable Log Rotation

**File Configuration field:**

nLogEnableRotate

**Description:**

Enables the log rotation post-processing subsystem for this File Action.

### Maximum wait time for log rotation

**File Configuration field:**

nLogRotateMaxWait

**Description:**

Maximum Wait time when log rotation is processed within the Queue Engine.

### Maximum number of rotated log files to keep

**File Configuration field:**

nLogRotateMaxFiles

**Description:**

Maximum number of rotated log file generations to keep. If set to 0, log files will not be renamed into numbered generations but can still be processed by Rotate Post Processing (for example compression or backup) along with the Rotate Conditions.

### Rotate Conditions

### Rotate each time a file is closed

**File Configuration field:**

nLogRotateOnClose

**Description:**

When a file is closed (Timeout for example), log rotation will be done.

### Do not rotate files on Shutdown

**File Configuration field:**

nLogDoNotRotateOnShutdown

**Description:**

Do not rotate log files if service is stopped even with “Rotate each time a file is closed” enabled.

Detached log rotation work that was already queued or already active in the background worker is still recovered on the next startup. This setting only suppresses starting an additional rotate-on-close operation because the service is shutting down.

### Rotate if this filesize limit is being reached

**File Configuration field:**

nLogRotateOnSizeLimit

**Description:**

Enable log rotation if a configured file size is reached.

### Filesize limit (KB)

**File Configuration field:**

nLogRotateSizeLimit

**Description:**

The actual file size in KB for “Rotate if this filesize limit is being reached”.

### Enable time based rotation

**File Configuration field:**

nLogEnableRotateTimeout

**Description:**

Enable time based log rotation.

### Rotate log files older than

**File Configuration field:**

nLogRotateTimeout

**Description:**

Sets the maximum file age before a logfile is being rotated when “Enable time based rotation” is enabled.

### Enable rotation by time of the day

**File Configuration field:**

nLogEnableRotateTimeOfTheDay

**Description:**

Rotate this file at this time (hour:minute) and the checked day/days.

### Rotate PostProcessing

### Compress File After log rotation

**File Configuration field:**

nLogZipAfterRotate

**Description:**

Enable file compression after log rotation.

Compression runs in the asynchronous log rotation worker after the live file has already been detached. If graceful shutdown happens while compression is still running, the unfinished detached work is resumed on the next startup instead of being abandoned.

### Compression Format

**File Configuration field:**

nLogZipAfterRotateFormat

**Description:**

It is possible to compress to ZIP or GZIP format.

### Compression Level

**File Configuration field:**

nLogZipCompressionLevel

**Description:**

There are different levels that can be selected:

- Best Speed
- Low Compression
- Normal Compression
- Best Compression

### Move file after log rotation

**File Configuration field:**

nLogMoveAfterRotate

**Description:**

Move logfile after rotation & compression.

Move-after-rotate uses the same asynchronous recovery model as compression. If shutdown reaches the configured worker wait timeout before the move finishes, the detached work is resumed on the next startup.

### Target directory

**File Configuration field:**

szLogMoveAfterRotatePath

**Description:**

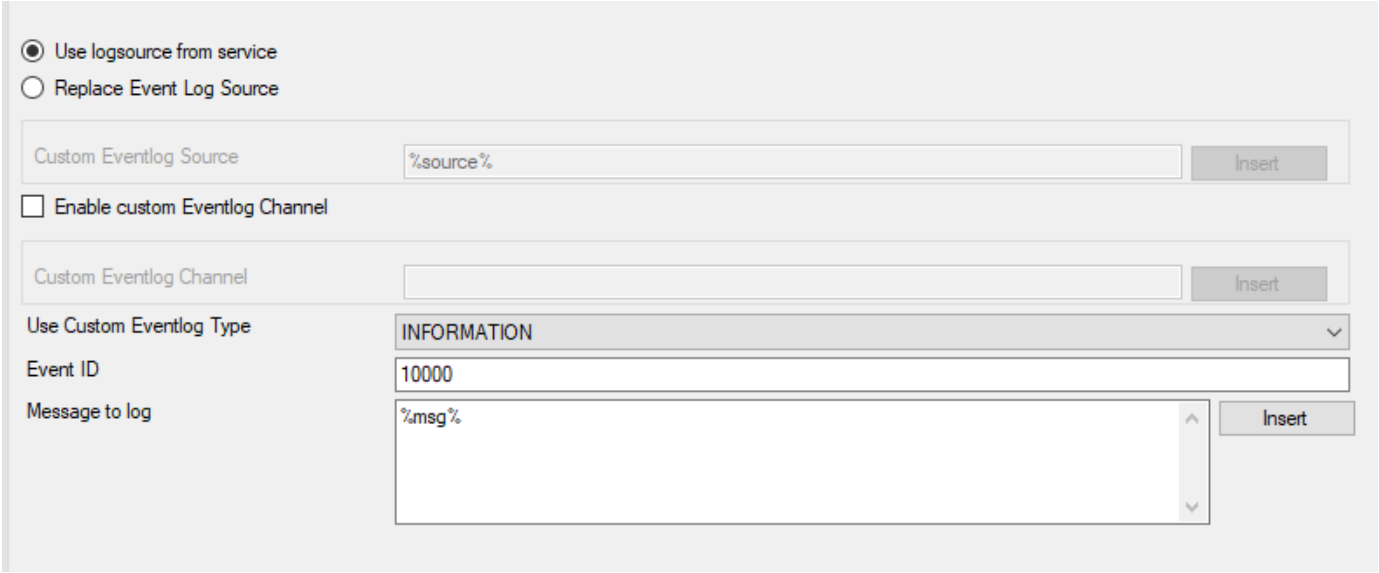
Location where to move the logfile after rotation & compression.

For an overview of shutdown, restart, reload, and dynamic-filename guarantees, see `../shared/faq/log-rotation-guarantees`.

## Forwarding and notification actions

### Event Log Options

This tab is used to configure the logging to the Windows Event Log. It is primarily included for legacy purposes.




- Action - EventLog*

#### Use logsource from service

**File Configuration field:**

bUseCustomEventLog = 0

**Description:**

Takes the service name as logsource for the log entry. This option is enabled by default.

#### Replace Event Log Source

**File Configuration field:**

bUseCustomEventLog = 1

**Description:**

If checked, a special mapping mechanism is activated. In this mode, the Windows event source is set to the IP address of the system sending the Syslog message. In addition, the ID is set to syslog facility. This mode helps to quickly gather information about the system state in Windows event viewer.

**However, this mode has its drawbacks.** Effectively, we are writing invalid event source information to the event log. This does not harm any application, but Windows event viewer will try to locate the matching message libraries. Of course, this is impossible. As such, event viewer will warn the user that the message library could not be found. Nevertheless, it will display the complete logged message. This happens only in detail view.

Users should fully understand the implications of this mapping mechanism for their environment before turning this option on.

#### Custom Event Log Source

**File Configuration field:**

szCustomSource

**Description:**

EventSource is now fully configurable with all possibilities the property engine gives you. Please note that content of this field can be configured. event properties are described in the property replacer section.

#### Enable custom Eventlog Channel

**File Configuration field:**

bUseCustomEventLog

**Description:**

If enabled, a custom event log channel will be used instead of application.

### Custom Eventlog Channel

**File Configuration field:**

szCustomEventLog

**Description:**

The custom Eventlog channel to be used instead of application. Will be automatically created if the channel does not exist.

### Use Custom Eventlog Type

**File Configuration field:**

nEventType

- 0 = EVENTLOG_SUCCESS (Information event)
- 1 = EVENTLOG_ERROR_TYPE (Error event)
- 2 = EVENTLOG_WARNING_TYPE (Warning event)
- 4 = EVENTLOG_INFORMATION_TYPE (Information event)
- 8 = EVENTLOG_AUDIT_SUCCESS (Success Audit event)
- 16 = EVENTLOG_AUDIT_FAILURE (Failure Audit event)

**Description:**

The type – or severity – this log entry is written with. Select from the available Windows system values.

### EventID

**File Configuration field:**

nEventID

**Description:**

The ID to be used when writing to the event log. Different IDs can be used to provide other processes with a consistent interface to specific messages. WinSyslog does not restrict the IDs that can be used. However, if an ID is written that is not registered with the operating system, Windows event viewer places an error message pointing this out before the actual message text. To avoid this text, event IDs 10,000 to 10,100 have been registered with the OS. We highly recommend that these IDs should be used for all custom messages. IDs below 10,000 should not be used as they might potentially interfere with events generated by WinSyslog 3.0 itself.

### Message to Log

**File Configuration field:**

szMessagecontent

**Description:**

It is the message which will be logged into the Windows Event Log. It is fully configurable what is logged into the Eventlog.

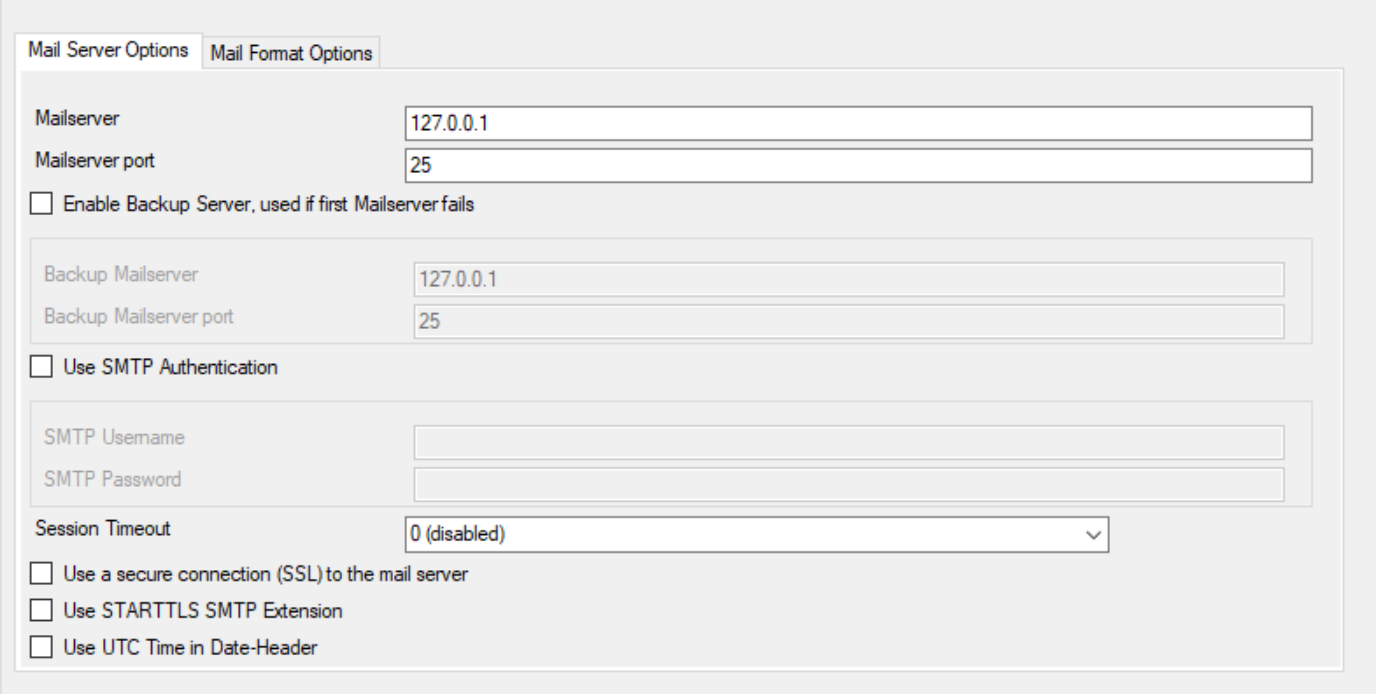
**Insert Menu entry allows you to add replacement characters e.g. ``%msg%`` - you can write the actual message of an event into the Windows Event Log.**

Please note that the message content of the message field can be configured. event properties are described in the property replacer section.

### Send Email

This tab is used to configure mail (SMTP) parameters. These are the basic parameters for email forwarding. They need to be configured correctly, if mail message should be sent by the service.

### Mail Server Options




- Action - Send Email - Mail Server Options*

### Mailserver

**File Configuration field:**

szSMTPServer

**Description:**

This is the Name or IP address of the mail server to be used for forwarding messages. You can either use an IPv4, an IPv6 Address, or a Hostname that resolves to an IPv4 or IPv6 Address. Please note that this server must be able to relay messages if the recipient is not hosted at this server. Be sure to contact your mail server's administrator if in doubt on this issue.

The service expects to talk to a standard SMTP mail server. Message relaying to the final destination must be permitted.

### Mailserver port

**File Configuration field:**

nSMTPPort

**Description:**

Port the mail server is to be contacted at. Usually, this is 25. It might, however, be changed in your system. Then, specify the port your mail server uses. If in doubt, try the default of 25 - or contact your mail server administrator.

### Enable Backup Server, used if first Mailserver fails

**File Configuration field:**

nEnableBackupServer

**Description:**

When enabled, you can configure a second Mailserver that will be used if the regular Mailserver is not available/accessible.

### Backup Mailserver

**File Configuration field:**

szSMTPServerBackup

**Description:**

In case that the connection to the main configured mail server cannot be established, the backup mail server is tried. Note that an error is only generated, if the connection to the backup server fails as well.

### Backup Mailserver port

**File Configuration field:**

nSMTPPortBackup

**Description:**

Port the mail server is to be contacted at. Usually, this is 25. It might, however, be changed in your system. Then, specify the port your mail server uses. If in doubt, try the default of 25 - or contact your mail server administrator.

### Use SMTP Authentication

**File Configuration field:**

nUseSMTPAuth

**Description:**

Select an authentication mode offered by your SMTP server: **None** or **Username and password (LOGIN)**. Older builds show a checkbox: unchecked means none; checked means LOGIN. For LOGIN, enter the username and password supplied by your mail server administrator.

### Session Timeout

**File Configuration field:**

nTimeoutValue

**Description:**

This option controls if multiple rapidly incoming messages should be combined to a single email message. The SMTP session with the server is held open for the specified timeout period. Please note that the period is specified in milliseconds, not seconds.

If a new event arrives within the specified timeout period, that event will be included in the same email message as the previous one. Then, the timeout is re-started. As such, any events coming in within successive timeout periods will be combined in a single mail.

This is most appropriate when large burst of messages are expected and these should be combined in few mail messages. Otherwise, multiple mail messages can easily overflow the administrator’s mailbox.

The session timeout is user configurable between 1 and 2147483647 milliseconds (32bit integer) or different pre-set values. Larger values are not supported as they probably affect the SMTP server performance and can lead to unpredictable results.

The session timeout of zero milliseconds has a special meaning: if it is selected, every event will be sent in a separate message, no matter how fast two messages occur after each other.

### Use a secure connection (SSL) to the mail server

**File Configuration field:**

nUseSSL

**Description:**

This option enables SSL-secured traffic to the mail server. Please note, that this only works, if the receiving mail server supports SSL-secured transmission of emails.

### Use STARTTLS SMTP Extension

**File Configuration field:**

nUseStartTLSMethod

**Description:**

Enable the STARTTLS SMTP extension when the mail server supports it. The server begins with a normal SMTP connection and upgrades it to a TLS-secured connection after the STARTTLS command.

### Use UTC Time in Date-Header

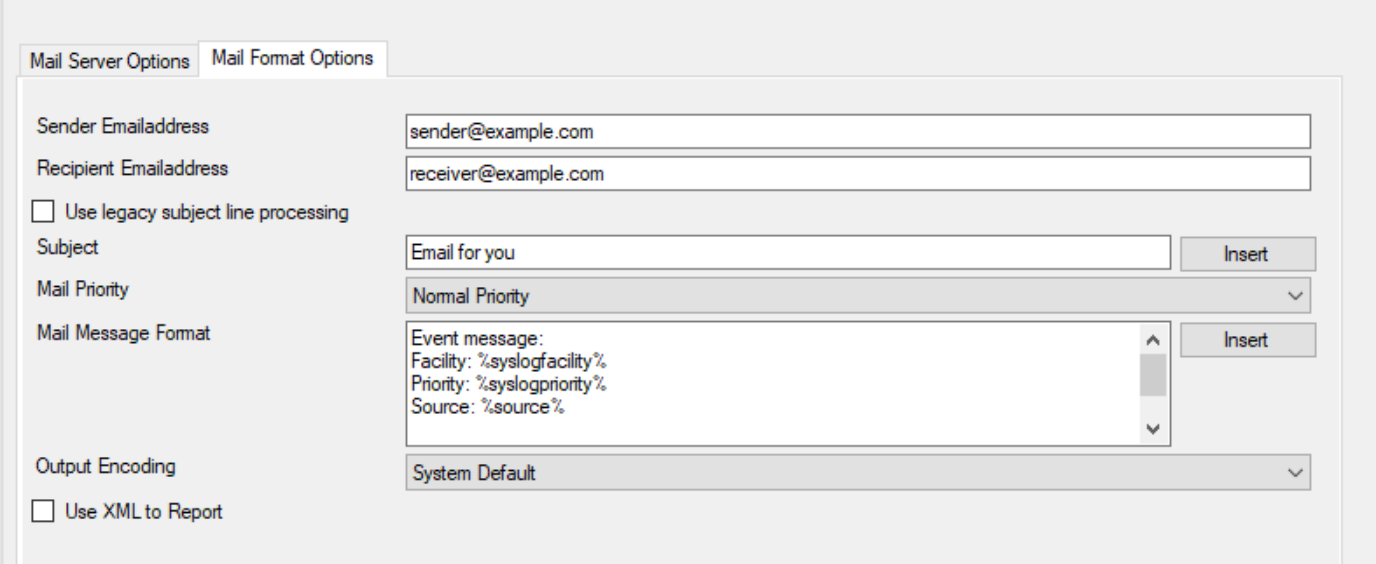
**File Configuration field:**

nUseUTCTimeStamp

**Description:**

Some Email Readers do not support UTC time in date-headers. Therefore here is a switch to turn the UTC time on or off.

### Mail Format Options




- Action - Send Email - Mail Format Options*

### Sender email address

**File Configuration field:**

szSMTPSender

**Description:**

Email address used as the sender address for outgoing messages. In order for your SMTP server to accept it, it probably must be a valid address.

### Recipient email address

**File Configuration field:**

szSMTPRecipient

**Description:**

The recipient emails are addressed to. To send a message to multiple recipients, enter all recipient's email addresses in this field. Separate addressed by spaces, semicolons, or commas

(e.g. "receiver1@example.com, receiver2@example.com"). Alternatively, you can use a single email address and define a distribution list in your mail software. The distribution list approach is best if the recipients frequently change or there is a large number of them. Multiple recipients are also supported. They can be delimited by space, comma, or semicolon.

### Use legacy subject line processing

**File Configuration field:**

nUseLegacySubjectProcessing

**Description:**

This checkbox specifies which type of subject line processing will be done. If it is checked, the old-style processing using single character replacement sequences is applied. If it is left unchecked, the far more powerful event property based method is used.

In legacy mode, the following replacement characters are recognized inside the subject line:

%s IP address or name (depending on the “resolve hostnames” setting) of the source system that sent the message.

%f Numeric facility code of the received message

%p Numeric priority code of the received message

%m the message itself. Please note: this is the complete message text and can be rather lengthy. As such, it is most probably subject to truncation. If that occurs, all other information after the %m replacement character is also truncated. As such, we strongly recommend using the %m replacement at the end of the subject line only.

%% It represents a single % sign. As an example, you may have the subject line set to `Event from %s: "m"` and enabled legacy processing. If a message `This is a test` were received from `172.16.0.1`, the resulting email subject would read: `Event from 172.16.0.1: This is a test`

In non-legacy mode, the Property Replacer can be used. With it, you can include any property from the event message and also modify it. Please visit the Property Replacer documentation for details.

As an example, in non-legacy mode, you can set the subject line to `Mesg: '%msg:1:15%' From: %fromhost%`. If the message `This is a lengthy test message` were received from `172.16.0.1`, the resulting email subject would read: `Mesg: 'This is a lengt' From: 172.16.0.1`. Please note that the message is truncated because you only extracted the first 15 characters from the message text (position 1 to 15).

### Subject

**File Configuration field:**

szSMTPSubject

**Description:**

Subject line to be used for outgoing emails and it is used for each message sent. It can contain replacement characters or “Event Properties” to customize it with event details. This is especially useful when sending email to cellular phones or pagers, which often display only the subject line and not the actual message body. The subject line – after expansion of the any replacement sequences – can hold a maximum of 255 characters. Characters beyond this will be truncated. Please note that many email systems impose a more strict limit and truncation may occur before the 255-character limit. It is advisable to limit the subject line length to 80 characters or less.

The mail body will also include full event information, including the source system, facility, priority, and actual message text as well as any other information that came with this event. As there is no size limitation for message bodies, the body always contains the full message received (except otherwise configured – see below).

Please note that Insert Menu entry allows you to add replacement characters e.g. `%msg%` - you can send out the actual message of an event in the subject line.

There will be one email for each received message. Email delivery is meant for urgent notifications and actions (e. g. calling pagers and such). It is not meant to provide an email report.

Please note that The message content of the Message field can be configured. Event properties are described in the property replacer section.

### Mail Priority

**File Configuration field:**

nMailPriority

- 0 = low
- 1 = Default
- 2 = High

**Description:**

Here you can adjust the priority with which the mail will be sent. You can choose between “low”, “normal”, and “high” priority. With this you can give your setup some complexity, being able to send some events as “important” and others with less importance.

### Mail Message Format

**File Configuration field:**

szSMTPBody

**Description:**

This is the format of the message body. Properties from the event can be included by using the Property Replacer. Please note that the message body is only sent if “Include Message/Event in Email Body” is checked.

### Output Encoding

**File Configuration field:**

nOutputEncoding

**Description:**

Determines the character encoding mode.

### Use XML to Report

**File Configuration field:**

nUseXMLtoReport

**Description:**

If checked, the received event will be included in XML format in the mail. If so, the event will include all information, like the original timestamp, the facility, priority etc. XML format is especially useful if the mail is sent to an automated system, which will then parse the message.

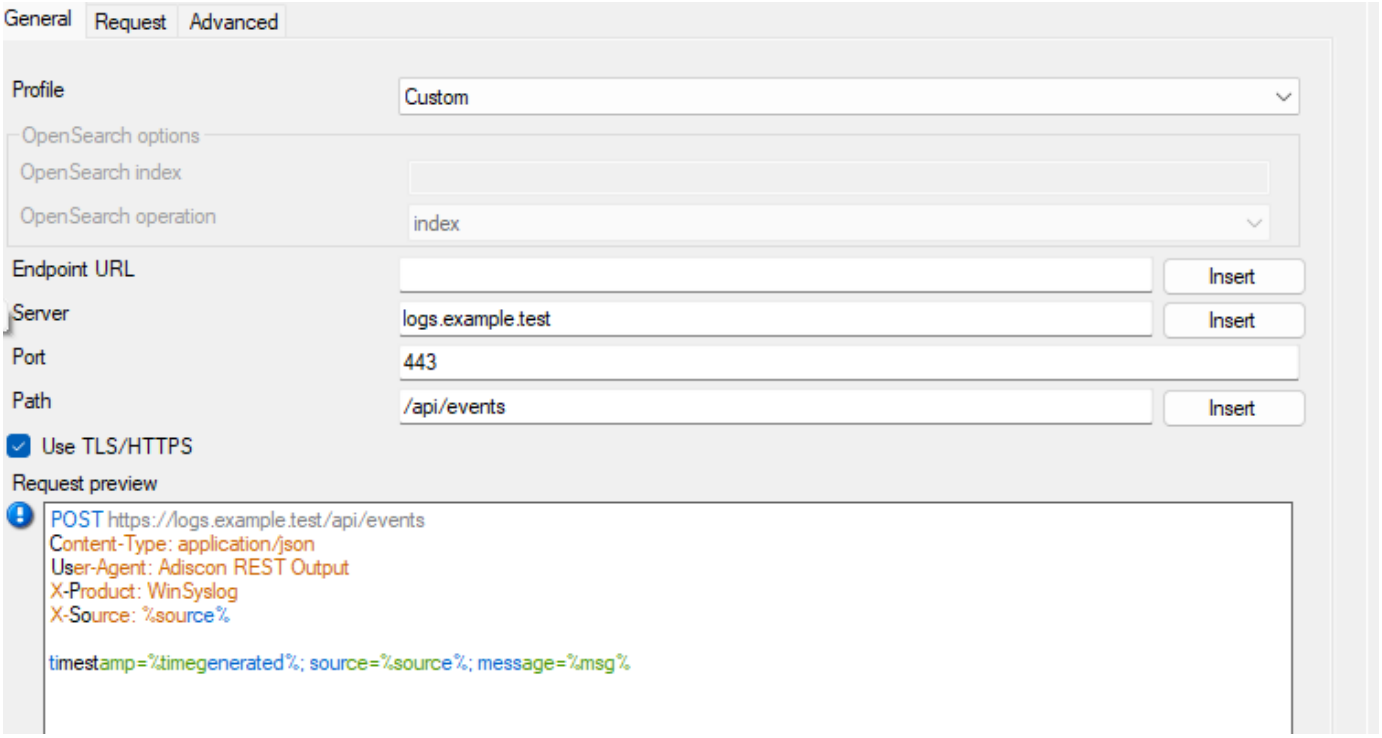
If unchecked, just the plain text message will be included in the mail. This format is more readable for a human reader.

### HTTP REST Output

The **HTTP REST Output** action forwards matched events or messages to an HTTP or HTTPS endpoint. It is available on WinSyslog, EventReporter, and WinSyslog starting with **26.07** (action type **1032**, `RestOutput`).

Use this action for webhooks, REST APIs, and OpenSearch bulk indexing. It is distinct from the older **HTTP Request** action, which performs a simpler HTTP call.

*Action - HTTP REST Output - General*




#### General tab

#### Profile

**File Configuration field:**

szProfile

**Description:**

Selects the request profile:

- **Custom** — full control over endpoint, method, headers, and body template.
- **OpenSearch bulk** — preconfigured layout for OpenSearch bulk API uploads.

#### Endpoint

**File Configuration field:**

szEndpoint

**Description:**

Optional full endpoint URL. When set, it can override separate server, path, and port fields depending on profile behavior. A preview shows the resolved URL in the client.

#### Server

**File Configuration field:**

szServer

**Description:**

Host name or IP address of the REST endpoint.

### Path

**File Configuration field:**

szPath

**Description:**

URL path (default `/`).

### Port

**File Configuration field:**

nPort

**Description:**

TCP port (`0` uses the default for HTTP or HTTPS).

### Use TLS

**File Configuration field:**

bUseTLS

**Description:**

Use HTTPS instead of HTTP.

### Ignore TLS certificate errors

**File Configuration field:**

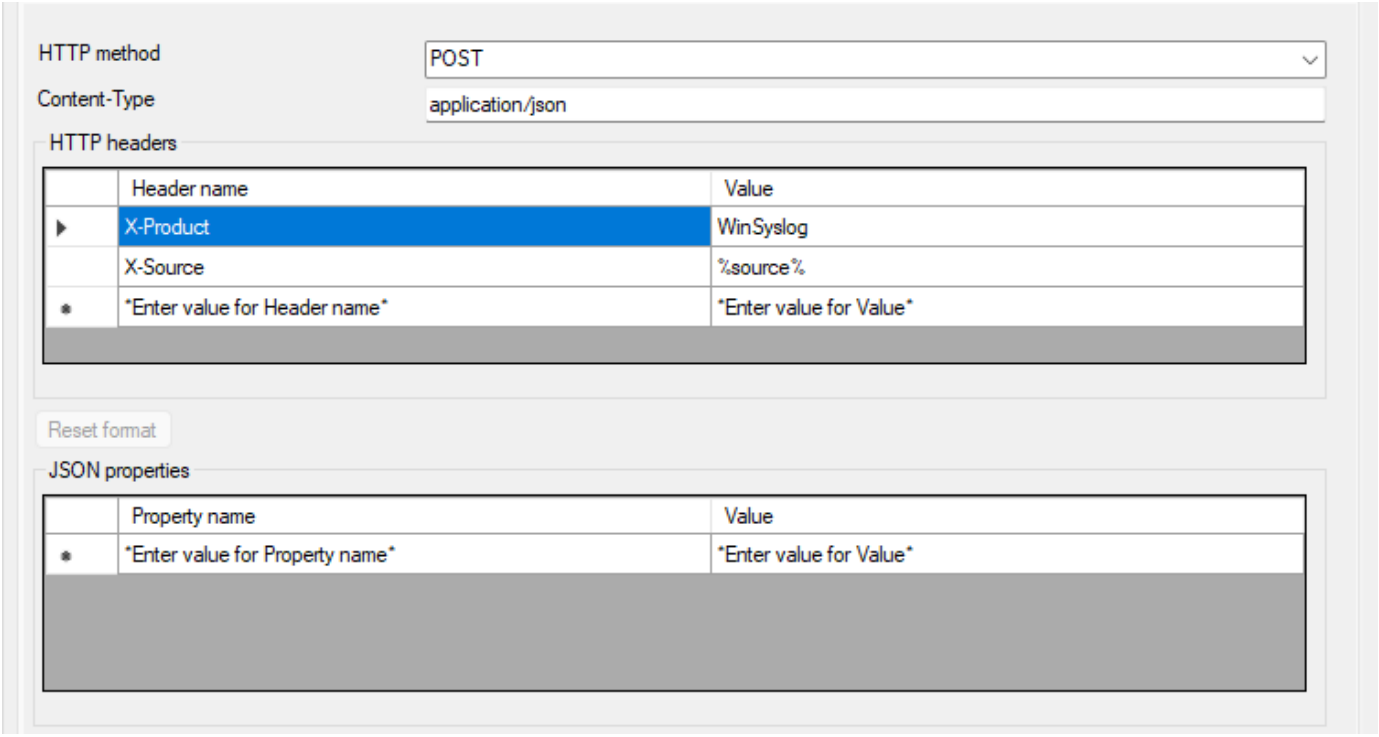
bIgnoreTlsCertificateErrors

**Description:**

Accept server certificates even when validation fails. Use only in controlled test environments.

### Request tab

*Action - HTTP REST Output - Request*



### Method

**File Configuration field:**

szMethod

**Description:**

HTTP method: `POST`, `PUT`, `PATCH`, `GET`, or `DELETE`.

### Content-Type

**File Configuration field:**

szContentType

**Description:**

Value for the `Content-Type` header (for example `application/json`).

### HTTP headers

**File Configuration field:**

szHttpHeaders

**Description:**

Additional headers as configured in the client header list.

### Request template

**File Configuration field:**

szRequestTemplate

**Description:**

Body template for the HTTP request. Property replacers can embed event fields. The client supports mapping JSON properties to values for structured payloads.

### OpenSearch bulk profile

When **OpenSearch bulk** is selected:

**File Configuration field:**

szOpenSearchIndex

**Description:**

Target index name.

**File Configuration field:**

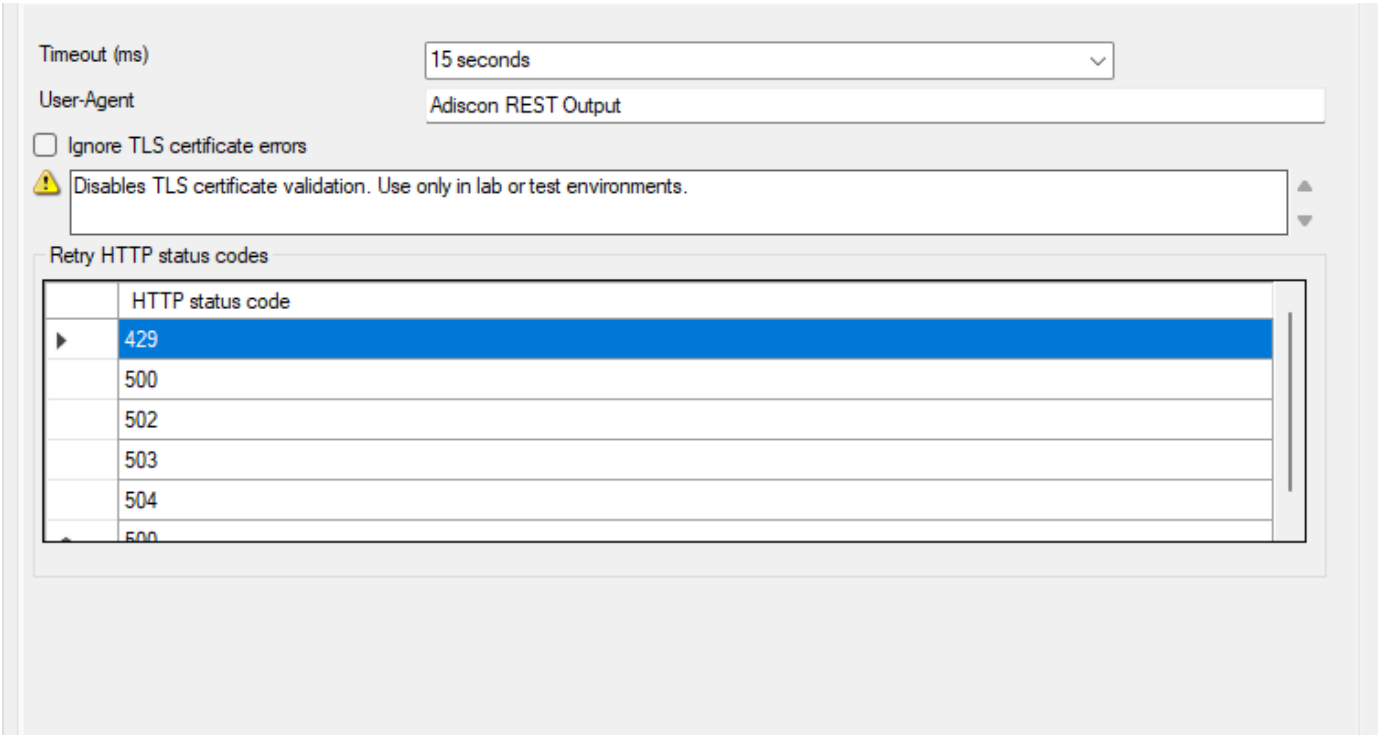
szOpenSearchOperation

**Description:**

Bulk operation type: `index` or `create`.

### Advanced tab

*Action - HTTP REST Output - Advanced*




### Timeout

**File Configuration field:**

nTimeoutMs

**Description:**

Maximum time in milliseconds to wait for the HTTP response (default 15000).

### User-Agent

**File Configuration field:**

szUserAgent

**Description:**

`User-Agent` header sent with the request (default `Adiscon REST Output`).

### Retry HTTP status codes

**File Configuration field:**

szRetryHttpStatusCodes

**Description:**

Comma-separated list of HTTP status codes that trigger a retry (default `429,500,502,503,504`). Individual codes can also be configured in the retry list in the client.

### Related information

- HTTP Request
- Accessing Properties

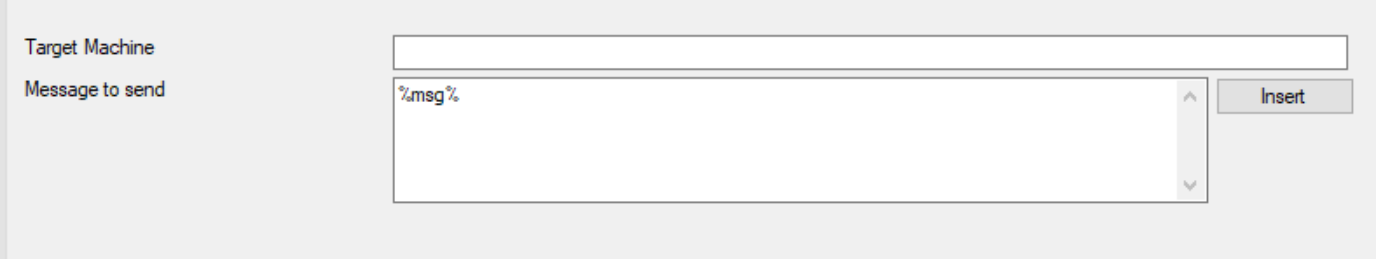
### Net Send

## Warning

**Removed from configuration client (26.07):** You cannot create new Net Send actions in WinSyslog, EventReporter, or WinSyslog from the **26.07** configuration client onward. Existing configurations that still contain Net Send may load with a skip or warning; use **Send Email**, HTTP REST Output, or **Start Program** for alerting instead.

**Deprecated:** The Windows Messenger service (`net send` pop-up messages) is not available by default on modern Windows versions and may be blocked or unavailable in managed environments.

With the “Net Send” action, short alert messages can be sent via the Windows “net send” facility. These messages are delivered on a best-effort basis. If the recipient can be reached, they will pop up in a message box on the recipient’s machine. If the recipient cannot be reached, they will simply be discarded. No buffering takes place. Consequently, the rule engine does not check if the message can be delivered. It will never flag an action to be in error due to a reported delivery problem with “net send”.




- Action - Net Send*

### Target Machine

**File Configuration field:**

szTarget

**Description:**

This is the Windows user name of the intended recipient, a NETBIOS machine name, or even an IP address (in the form of `10.1.1.1`). You can either use an IPv4, an IPv6 Address or a Hostname that resolves to an IPv4 or IPv6 Address.

### Message to send

**File Configuration field:**

szMessage

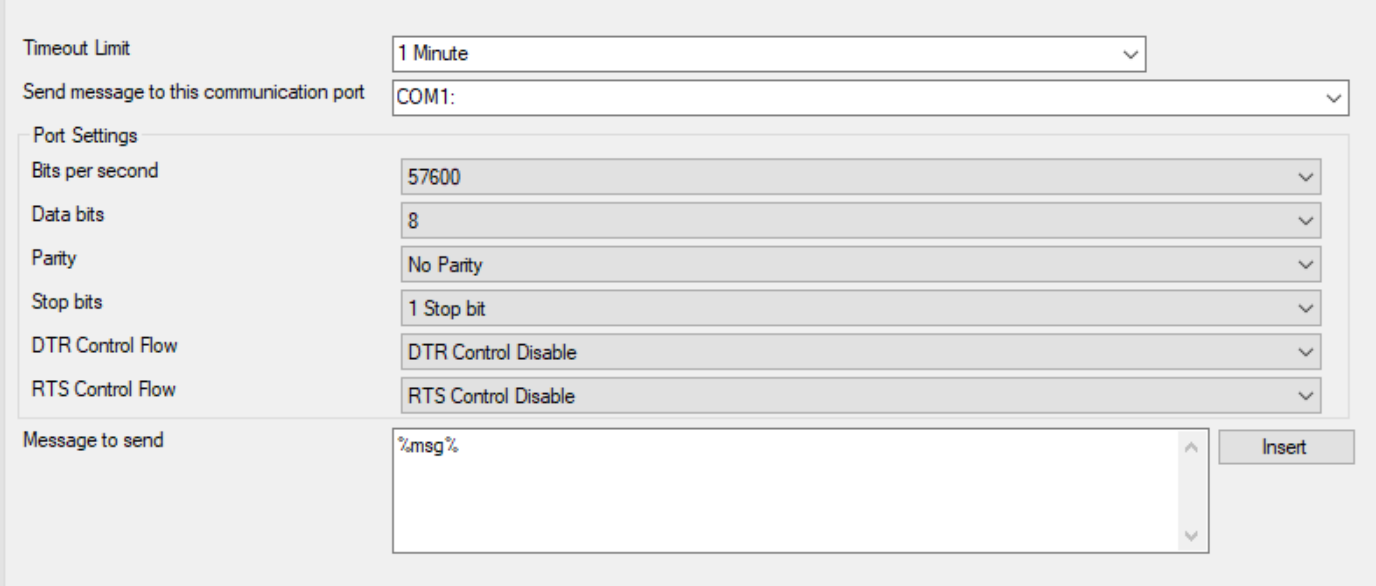
**Description:**

This is the message that is sent to the intended target.

Please note that the message content of the Message to send field can now be configured. event properties are described in the property replacer section.

### Send to Communications Port

This action allows you to send a string to an attached communications device, that is, it sends a message through a Serial Port.




- Action - Send to Communication Port*

### Timeout Limit

**File Configuration field:**

nTimeOutLimit

**Description:**

The maximum time allowed for the device to accept the message. If the message could not be sent within that period, the action is aborted. Depending on the device, it may be left in an unstable state.

### Send message to this communication port

**File Configuration field:**

szPortName

**Description:**

Specify the port to which your device is being attached. Typically, this should be one of the COMx: ports. The list box shows all ports that can be found on your local machine. You may need to adjust this to a different value, if you are configuring a remote machine.

1. MSFAX
2. COM1
3. COM2
4. COM3
5. COM4
6. FILE
7. LPT1
8. LPT2
9. LPT3
10. AVMISDN1
11. AVMISDN2
12. AVMISDN3

13 AVMISDN4
.
14 AVMISDN5
.
15 AVMISDN6
.
16 AVMISDN7
.
17 AVMISDN8
.
18 AVMISDN9
.

### Port Settings

**File Configuration field:**

szPortSettings

**Description:**

Use those settings that your device expects. Please consult your device manual if in doubt.

### Bits per Seconds

**File Configuration field:**

nBps

**Description:**

Bits per second can be 110 and go up to 256000, by default 57600 is selected.

### Databits

**File Configuration field:**

nDatabits

**Description:**

Databits defines that how many bits you want to send and receive to the communication port.

### Parity

**File Configuration field:**

nParity

**Description:**

With Parity you can configure the Parity scheme to be used. This can be one of the following values:

1. Even
2. Mark
3. No parity
4. Odd
5. Space

### Stop bits

**File Configuration field:**

nStopbits

**Description:**

You can configure the number of stop bits to be used. This can be one of the following values:

1. 1 stop bit

2. 1.5 stop bits
3. 2 stop bits

### DTR Control Flow

**File Configuration field:**

nDtsControl

**Description:**

DTR (data-terminal-ready) flow control. This member can be one of the following values:

1. DTR_CONTROL_DISABLE - Disables the DTR line when the device is opened and leaves it disabled.
2. DTR_CONTROL_ENABLE - Enables the DTR line when the device is opened and leaves it on.
3. DTR_CONTROL_HANDSHAKE - Enables DTR handshaking.

### RTS Control Flow

**File Configuration field:**

nRtsControl

**Description:**

RTS (request-to-send) flow control. This member can be one of the following values:

1. RTS_CONTROL_DISABLE - Disables the RTS line when the device is opened and leaves it disabled.
2. RTS_CONTROL_ENABLE - Enables the RTS line when the device is opened and leaves it on.
3. RTS_CONTROL_HANDSHAKE - Enables RTS handshaking. The driver raises the RTS line when the “type-ahead” (input) buffer is less than one-half full and lowers the RTS line when the buffer is more than three-quarters full.
4. RTS_CONTROL_TOGGLE - Specifies that the RTS line will be high if bytes are available for transmission. After all buffered bytes have been sent, the RTS line will be low.

### Message to Send

**File Configuration field:**

szMessage

**Description:**

This is the message that is to be sent to the device. You can enter text plainly and you can also include all properties from the current event. For example, if you have a serial audit printer and you would just plainly like to log arrived messages to that printer, you could use the string “%msg%%$CRLF%” to write the actual message arrived plus a CRLF (line feed) sequence to the printer.

Please note that the message content of the Message field can now be configured. event properties are described in the property replacer section.

### Send MSQueue

In order to use this Action, the “Microsoft Message Queue (MSMQ) Server” needs to be installed. This Action can be used to send a message into the Microsoft Message Queue.




- Action - Send MSQueue*

#### Server Computername/IP

**File Configuration field:**

szComputerName

**Description:**

Sets the computername or IP which contains the MSQueue you want to query. You can either use an IPv4, an IPv6 Address, or a Hostname that resolves to an IPv4 or IPv6 Address.

#### Queuename

**File Configuration field:**

szQueueName

**Description:**

Specify the Queuename into which you want to write.

#### Queue Priority

**File Configuration field:**

nMessagePriority

**Description:**

Configure or set the priority property here.

#### Queue Message Label

**File Configuration field:**

szQueueLable

**Description:**

Sets the Label text of a queue item.

#### Queue Message Body

**File Configuration field:**

szQueueBody

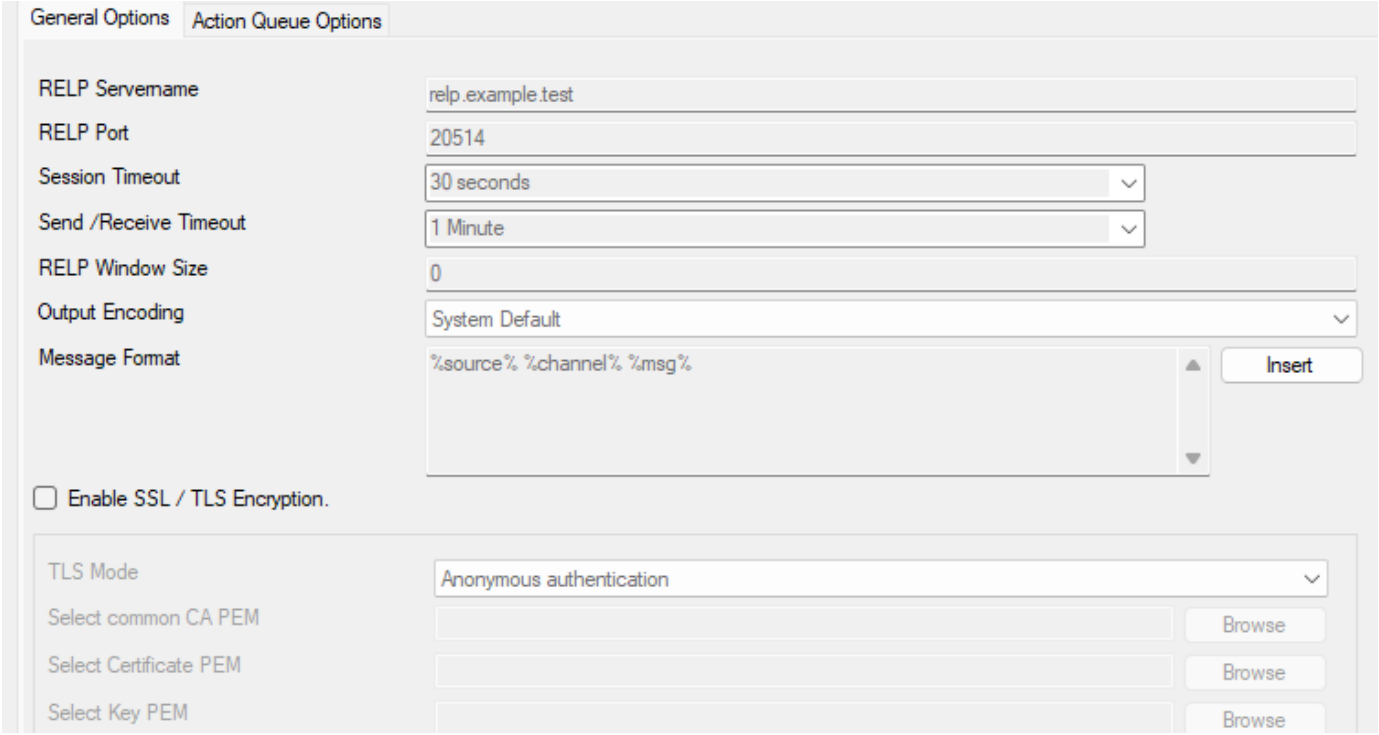
**Description:**

The text here will be set to the body of a queue item.

### Send RELP

This action is roughly equivalent to the “send syslog” action, except that it utilizes the new reliable event logging protocol (RELP) for message transmission. It can only be used together with a RELP-enabled receiver but then provides enhance reliability in the communications process.

Note that for reliable operation where messages should be preserved over a service shutdown, queue cache mode must be activated. This is because RELP guards only the transmission channel, but not local processing.




- Action - Send RELP General*

### RELP Servername

**File Configuration field:**

szSelpSendServer

**Description:**

This is the name or IP address of the system to which RELP messages should be sent to. You can either use an IPv4, an IPv6 Address, or a Hostname that resolves to an IPv4 or IPv6 Address.

### RELP Port

**File Configuration field:**

nSelpSendPort

**Description:**

The remote port on the RELP server to report to. If in doubt, please leave it at the default value of 20514, which is typically the RELP port. Different values are only required for special setups, for example in security sensitive areas.

Instead of the port number, a service name can be used. If so, that name is looked up via the socket service database functions.

### Session Timeout

**File Configuration field:**

nTimeOutSession

**Description:**

The maximum time a session to a SETP server is to be kept open.

### Send / Receive Timeout

**File Configuration field:**

nSendTimeOut

**Description:**

The maximum time a server waits for a response of a remote server. When the timeout expires without receiving a response, the connection is broken and (based on rule settings) being reestablished. This can be a useful option if the remote system drops connections for whatever reason AND the sender system is not notified about this (which, for example, can happen due to some firewall configurations).

### RELP Window Size

**File Configuration field:**

nSelpWindowSize

**Description:**

Configures the RELP send window size (0–4096). When greater than `1`, the value is passed to the RELP client library to control how many messages can be in flight before acknowledgment. Use `0` or `1` for default librelp behavior. Tune this when you need higher throughput on reliable links or stricter flow control on constrained links.

### Output Encoding

**File Configuration field:**

nOutputEncoding

**Description:**

Allows you to specify the character encoding for messages sent to the RELP server. The default setting is “System Default”, which uses the system’s default character encoding. Other common options include UTF-8, ASCII, and other standard encodings.

### Message format

**File Configuration field:**

szMessage

**Description:**

You can change the message format. By default the original message is forwarded.

Please note that the message content of the Message field can be configured. event properties are described in the property replacer section.

### Enable SSL / TLS Encryption

**File Configuration field:**

nUseSSL

**Description:**

If this option is enabled, the action will use SSL/TLS encryption for secure communication with the RELP server. When disabled, messages will be sent unencrypted. Note that if this option is enabled, the action will not be able to talk to a NON-SSL secured server.

### TLS Mode

**File Configuration field:**

nTLSMode

**Description:**

**Anonymous Authentication** Default option. This means that a default certificate will be used.

**Use Certificate** If this option is enable, you can specify your own certificate. For further authentication solutions, you will need to create your own certificates using OpenSSL Tools for example.

### Select common CA PEM

**File Configuration field:**

szTLSCAFile

**Description:**

Select the certificate from the common Certificate Authority (CA). The RELP Receiver should use the same CA.

### Select Certificate PEM

**File Configuration field:**

szTLSCertFile

**Description:**

Select the client certificate (PEM Format).

### Select Key PEM

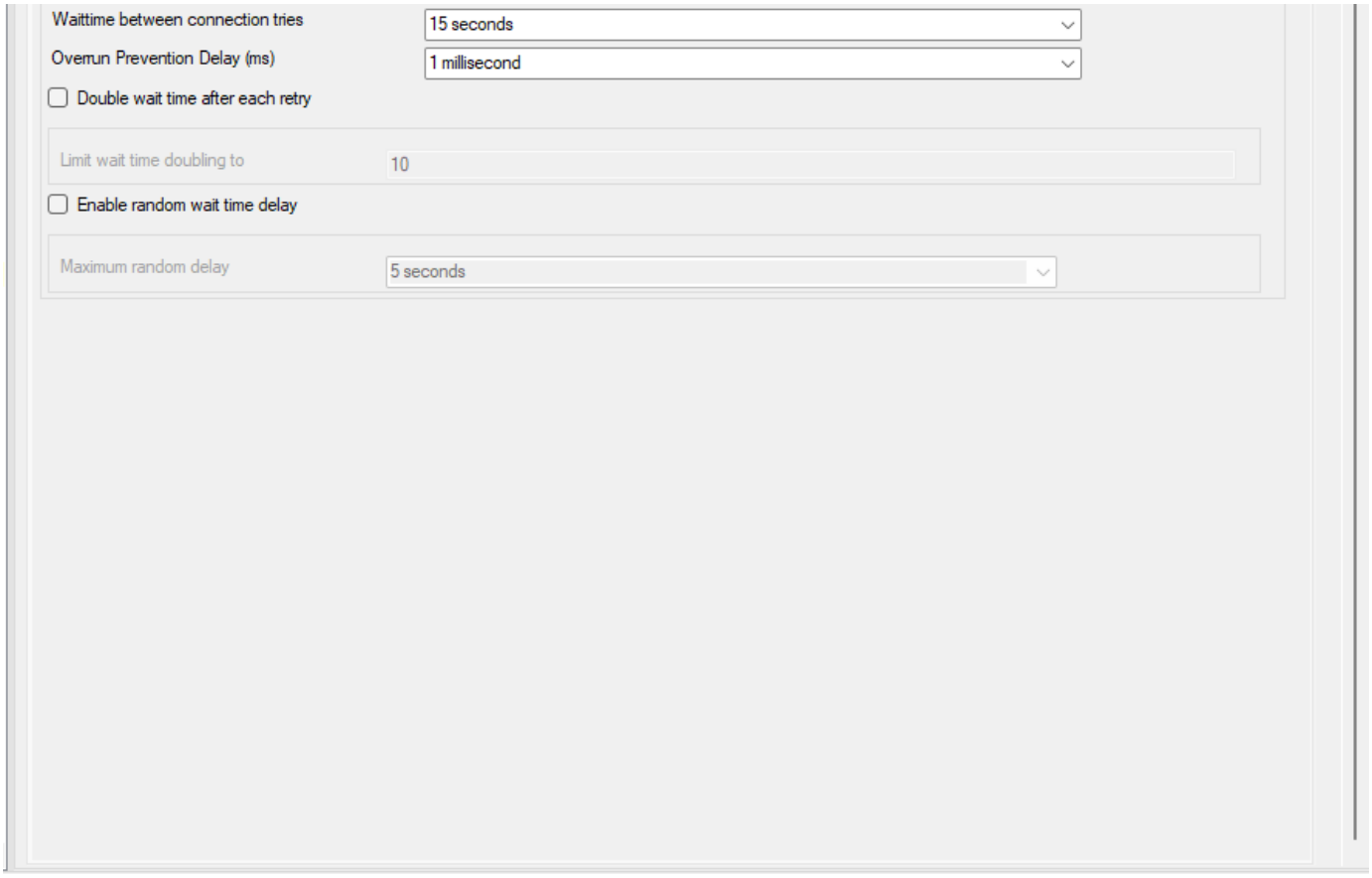
**File Configuration field:**

szTLSKeyFile

**Description:**

Select the keyfile for the client certificate (PEM Format).

### Action Queue Options




*WinSyslog - Send RELP Action Queue Options*

### Use Diskqueue if connection to Syslog server fails

**File Configuration field:**

nUseDiscQueue

**Description:**

Enable diskqueuing syslog messages after unexpected connection loss.

### Split files if this size is reached

**File Configuration field:**

nDiskQueueMaxFileSize

**Description:**

Files are split when they reach this size in bytes. The default is 10485760 (10 MB). The maximum per file is 2147483648 bytes (2 GB). The total diskqueue size is limited by free disk space.

### Diskqueue Directory

**File Configuration field:**

szDiskQueueDirectory

**Description:**

The directory where the queue files will be generated in. The queuefiles will be generated with a dynamic UUID bound to the action configuration.

### Waittime between connection tries

**File Configuration fields:**

nDiskCacheWait

**Description:**

The minimum waittime until the Syslog Action retries to establish a connection to the Syslog server after failure.

### Overrun Prevention Delay (ms)

**File Configuration field:**

nPreventOverrunDelay

**Description:**

When the Action is processing syslog cache files, an overrun prevention delay can be added to avoid flooding the target Syslog server.

### Double wait time after each retry

**File Configuration field:**

bCacheWaittimeDoubling

**Description:**

If enabled, the configured waittime is doubled after each try.

### Limit wait time doubling to

**File Configuration field:**

nCacheWaittimeDoublingTimes

**Description:**

How many times the wait time is doubled after a failed connection try. This is a count, not seconds. The range is 1 to 100, and the default is 10.

### Enable random wait time delay

**File Configuration field:**

bCacheRandomDelay

**Description:**

If enabled, a some random time will be added into the waittime delay. When using many syslog senders, this can avoid that all senders start sending cached syslog data to the Syslog server at the same time.

### Maximum random delay

**File Configuration field:**

nCacheRandomDelayTime

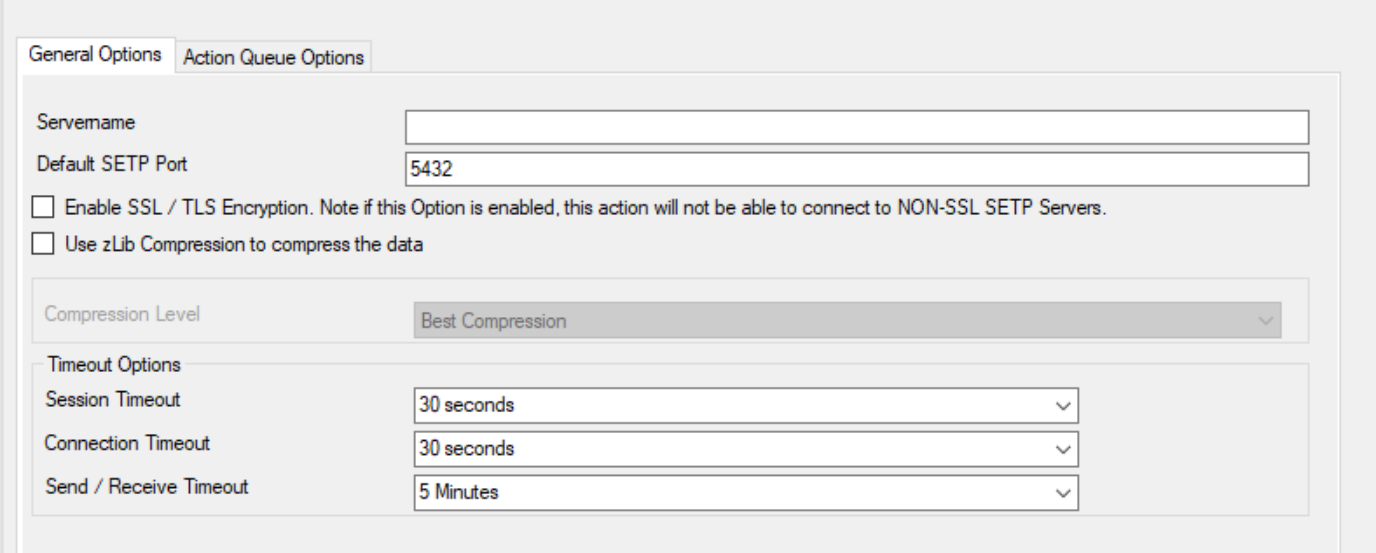
**Description:**

Maximum random delay time that will be added to the configured waittime if Enable random wait time delay is enabled.

### Send SETP

With the “Send SETP” action, messages can be sent to a SETP server.

In WinSyslog, the **Send SETP** action is a sender-side action. Receiving SETP events requires the separate `SETP Server` service.




- Action - Send SETP General*

### Servername

**File Configuration field:**

szServer

**Description:**

The product sends setp to the server/listener under this name. You can either use an IPv4, an IPv6 Address, or a Hostname that resolves to an IPv4 or IPv6 Address.

### Default SETP Port

**File Configuration field:**

nMIAPSendPort

**Description:**

The Send setp sends outgoing requests on this port. The default value is 5432. Set the port to 0 to use the system-supplied default value (which defaults to 5432 if not modified by a system administrator).

Instead of the port number, a service name can be used. If so, that name is looked up via the socket service database functions. The lookup is for protocol TCP.

**Please Note: The SETP port configured here must match the port configured at the listener side (for example, WinSyslog Enterprise edition). If they do not match, a Send SETP session cannot be initiated. The rule engine will log this to the Windows Event Log.**

### Enable SSL / TLS Encryption

**File Configuration field:**

nUseSSL

**Description:**

If this option is enabled then this action will be able to connect to SSL/TLS setp servers. Please make sure that you want this option to be enabled.

### Use zLib Compression to compress the data

**File Configuration field:**

nZlibComp

**Description:**

It enables zLib compression support. Note that the SETP receiver must have zLib Compression support and enabled, otherwise it does not work.

### Compression Level

**File Configuration field:**

nCompLevel

- 1 = Best Speed
- 3 = Low Compression
- 6 = Normal Compression
- 9 = Best Compression

**Description:**

Higher level results in better compression but slower performance.

### Session Timeout

**File Configuration field:**

nTimeOutSession

**Description:**

The maximum time a session to a SETP server is to be kept open.

### Connection Timeout

**File Configuration field:**

nConnectTimeOut

**Description:**

Maximum time a connection can take to connect or disconnect.

### Send / Receive Timeout

**File Configuration field:**

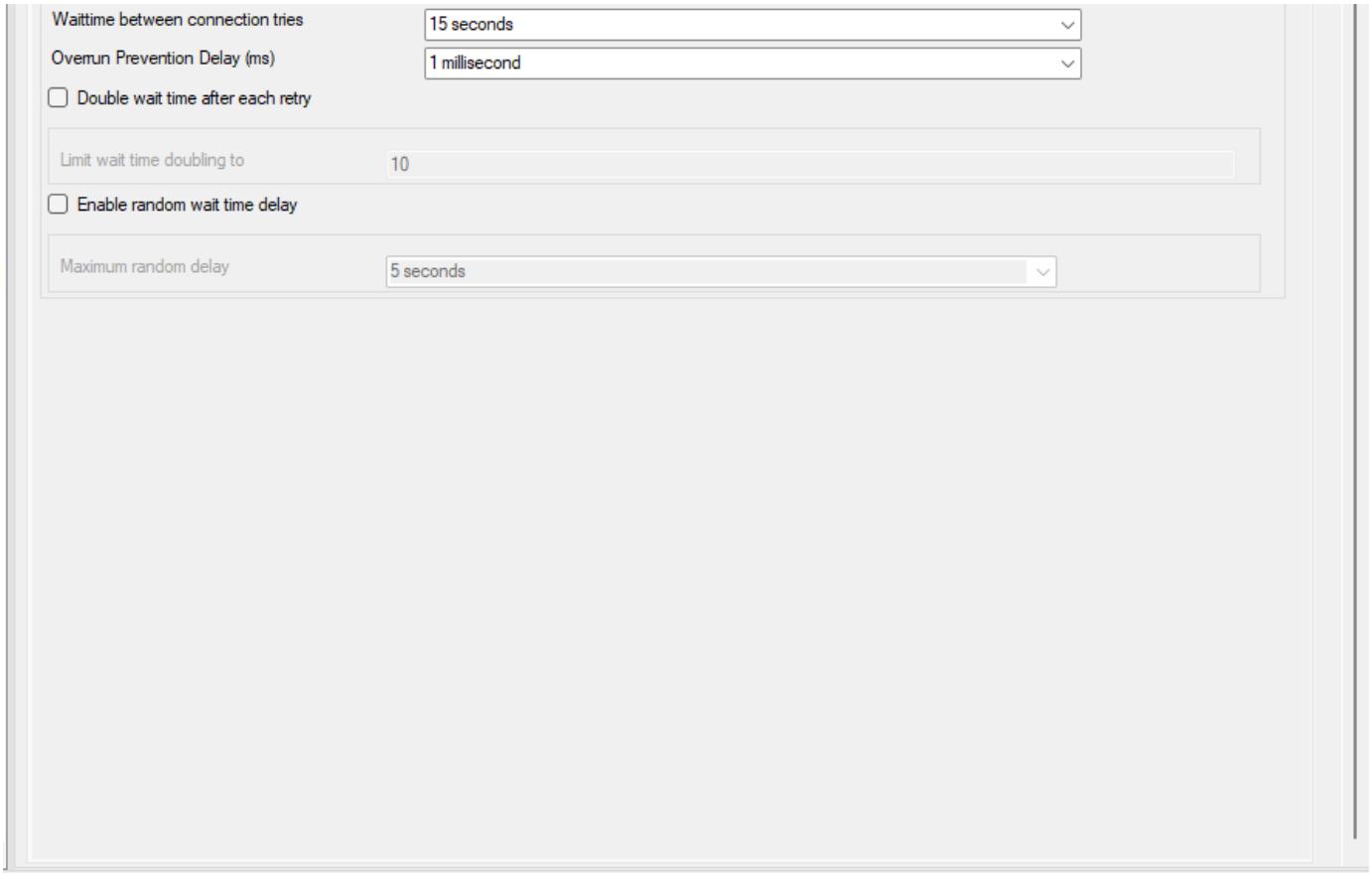
nSendRecvTimeOut

**Description:**

When sending or receiving data, this timeout applies.

**Please note:** If this option is enabled, this action is not be able to connect to NON-SSL SETP servers.

### Action Queue Options




*WinSyslog - Send SETP Action Queue Options*

### Use Diskqueue if connection to Syslog server fails

**File Configuration field:**

nUseDiscQueue

**Description:**

Enable diskqueuing syslog messages after unexpected connection loss.

### Split files if this size is reached

**File Configuration field:**

nDiskQueueMaxFileSize

**Description:**

Files are split when they reach this size in bytes. The default is 10485760 (10 MB). The maximum per file is 2147483648 bytes (2 GB). The total diskqueue size is limited by free disk space.

### Diskqueue Directory

**File Configuration field:**

szDiskQueueDirectory

**Description:**

The directory where the queue files will be generated in. The queuefiles will be generated with a dynamic UUID bound to the action configuration.

### Waittime between connection tries

**File Configuration fields:**

nDiskCacheWait

**Description:**

The minimum waittime until the Syslog Action retries to establish a connection to the Syslog server after failure.

### Overrun Prevention Delay (ms)

**File Configuration field:**

nPreventOverrunDelay

**Description:**

When the Action is processing syslog cache files, an overrun prevention delay can be added to avoid flooding the target Syslog server.

### Double wait time after each retry

**File Configuration field:**

bCacheWaittimeDoubling

**Description:**

If enabled, the configured waittime is doubled after each try.

### Limit wait time doubling to

**File Configuration field:**

nCacheWaittimeDoublingTimes

**Description:**

How many times the wait time is doubled after a failed connection try. This is a count, not seconds. The range is 1 to 100, and the default is 10.

### Enable random wait time delay

**File Configuration field:**

bCacheRandomDelay

**Description:**

If enabled, a some random time will be added into the waittime delay. When using many syslog senders, this can avoid that all senders start sending cached syslog data to the Syslog server at the same time.

### Maximum random delay

**File Configuration field:**

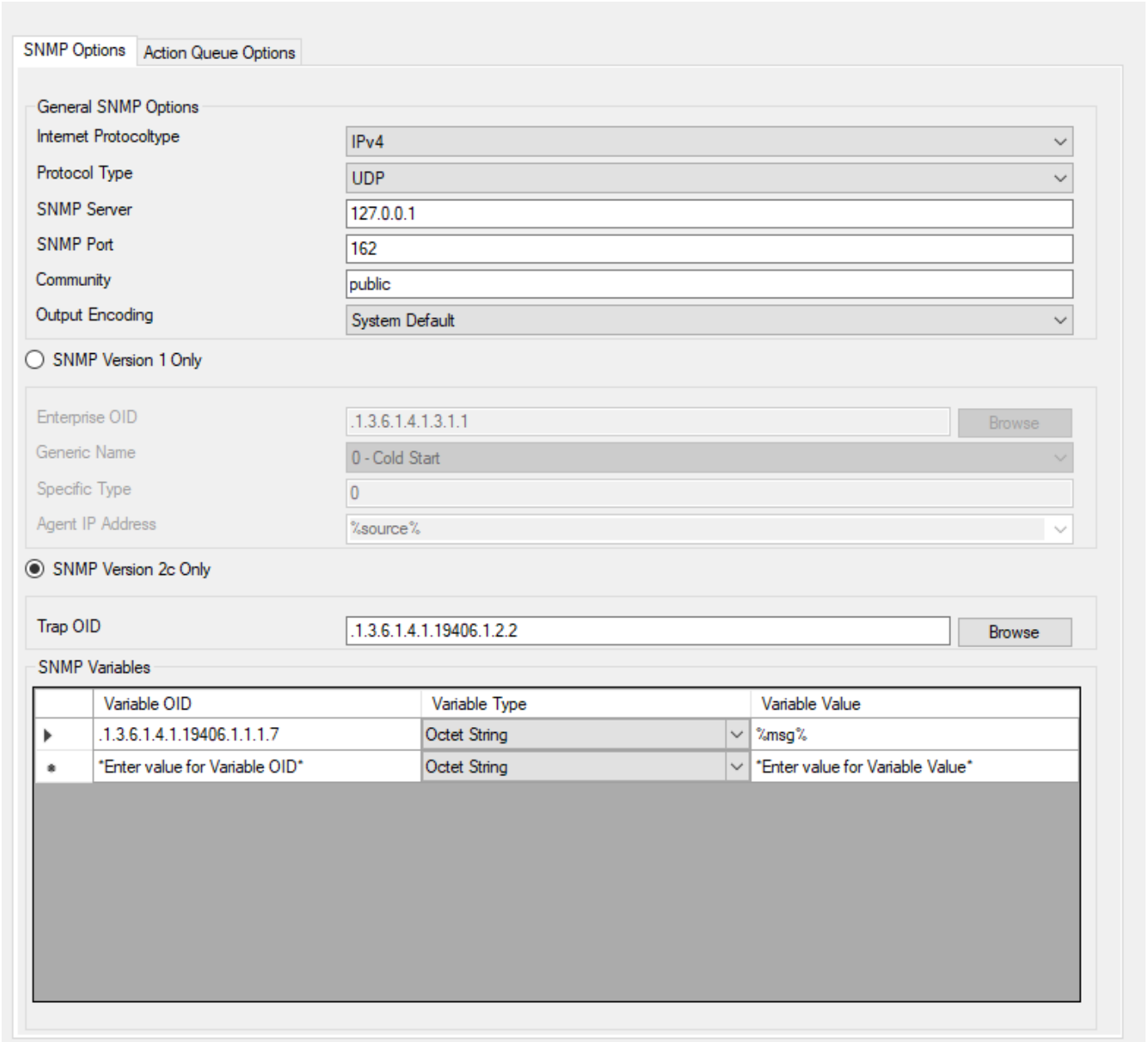
nCacheRandomDelayTime

**Description:**

Maximum random delay time that will be added to the configured waittime if Enable random wait time delay is enabled.

## Send SNMP Trap

### General Options




- Action - Send SNMP Trap Options*

**SNMP Version**

You can choose between SNMP Version 1 Only and SNMP Version 2c Only. Both options have different SNMP related configuration properties which need to be configured.

### Internet Protocoltype

**File Configuration field:**

nInetType

**Description:**

Select the desired protocol type. IPv4 and IPv6 are available. The IPv6 protocol needs to be properly installed in order to be used. Note that one Service can only handle IPv4 or IPv6, so if you want to use both protocols, you will need to create two separate services.

### Protocol Type

**File Configuration field:**

nProtocolType

**Description:**

You can select to listen on UDP or TCP protocol for SNMP Traps.

### SNMP Server

**File Configuration field:**

szAgent

**Description:**

Specify the agent that has to receive the SNMP trap. Please note that specifying a host name can cause the SMTP probe to fail if DNS name resolution fails (for example due to a failing DNS server). To avoid this, specify an IP address. You can either use an IPv4, an IPv6 Address or a Hostname that resolves to an IPv4 or IPv6 Address.

### SNMP Port

**File Configuration field:**

nPort

**Description:**

This port is to be probed. Please see your server's reference for the actual value to use. For example, mail servers typically listen to port 25 and web servers to 80.

### Community

**File Configuration field:**

szCommunity

**Description:**

Specify the SNMP community to which the messages belong too.

### SNMP V1 Specific Parameters

Under this group box you can see the parameters related to SNMP version 1.

### Enterprise OID

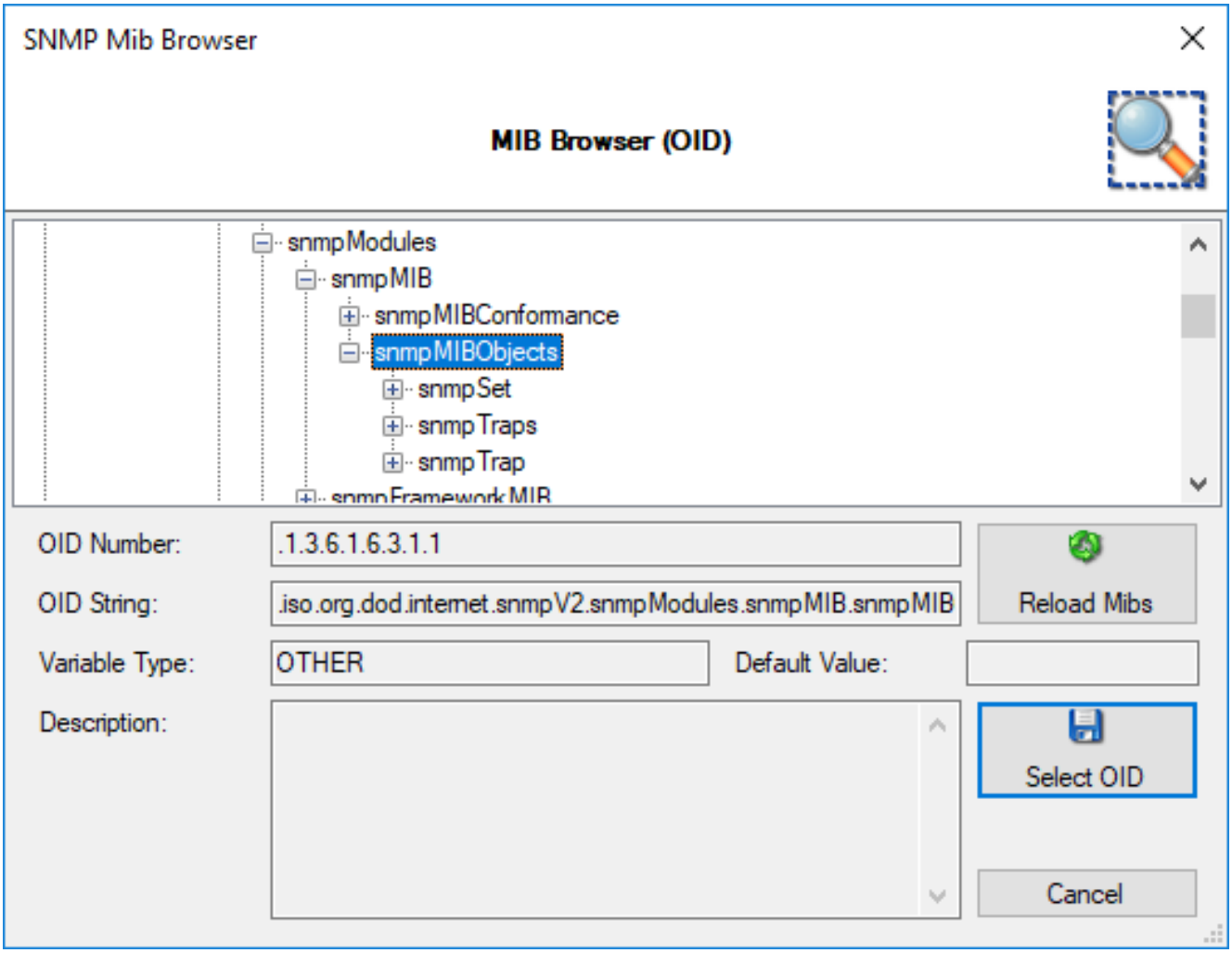
**File Configuration field:**

nInetType

**Description:**

It is also an optional value which can be used to specify a UserAgent that is send in the HTTP header.

Specify the enterprise object ID here. You can use Browse option to select your OID. If you click the Browse link, the screen similar to shown below is appeared:




MIB Browser

You can select your MIB here.

### Generic Name

**File Configuration field:**

nGenericName

- 0 - Cold Start
- 1 - Warm Start
- 2 - Link Down
- 3 - Link Up
- 4 - Authentication Failure
- 5 - EGP Neighbor Loss
- 6 - Enterprise Specific

**Description:**

You can specify the generic name of the trap which can be one of these: coldStart(0), warmStart(1), linkDown(2), linkUp(3), authenticationFailure(4), egpNeighborLoss(5), or enterpriseSpecific(6).

### Specific Type

**File Configuration field:**

nSpecificType

**Description:**

You can define an additional code for the trap. It is also an Integer value.

### Agent IP Address

**File Configuration field:**

szAgentIP

**Description:**

The SNMPv1 Agent Address field can be set to other IP Addresses here. Hostnames will automatically be resolved if possible. By default we are using the %source% property.

**SNMP Variables**

These are the variables to send in the SNMP Trap. If you know the trap codes, you can enter them manually, otherwise use the built-in SNMP MIB Browser.

### Variable OID

**File Configuration field:**

szVariableOID_[n]

**Description:**

OID of the SNMP Trap. Use the built-in SNMP MIB Browser for a list of known and available OIDs.

### Variable Type

**File Configuration field:**

nVariableType_[n]

- 1 = TYPE_OBJID
- 2 = TYPE_OCTETSTR
- 3 = TYPE_INTEGER
- 5 = TYPE_IPADDR
- 6 = TYPE_COUNTER
- 7 = TYPE_GAUGE
- 8 = TYPE_TIMETICKS
- 12 = TYPE_BITSTRING
- 14 = TYPE_UINTEGER
- 15 = TYPE_UNSIGNED32
- 16 = TYPE_INTEGER32

**Description:**

The variable type of the variable, usually OCTETSTRING or INTEGER. Depending on this type, the Variable value needs to be formatted correctly (Like for the type IPADDR).

### Variable Value

**File Configuration field:**

szVariableValue_[n]

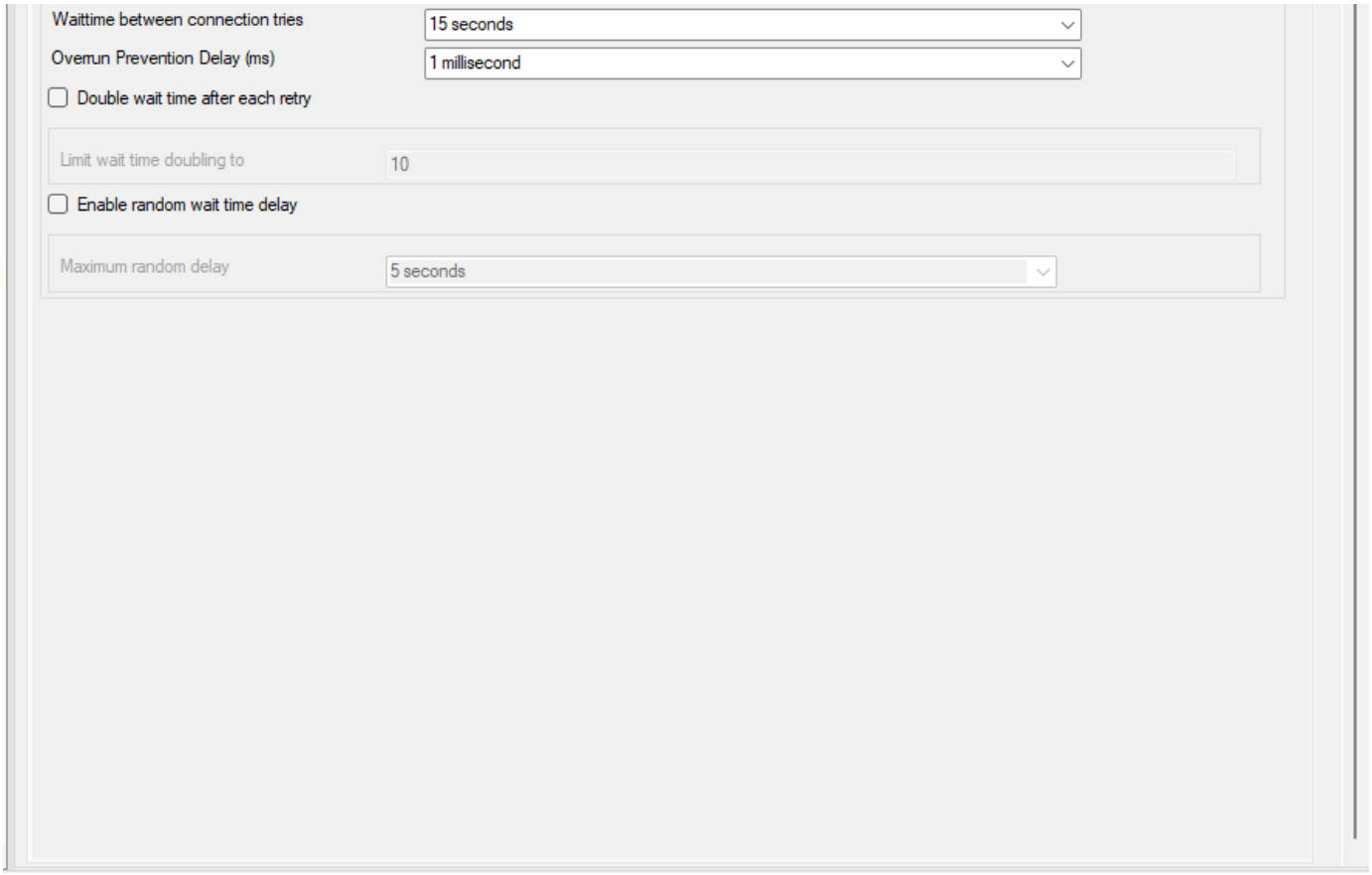
**Description:**

The value of the Variable. It needs to be formatted depending on the variable type.

**Please Note:**

**The “Send SNMP Trap”-Action is capable of sending all kinds of Traps. You can choose the whole variety of the MonitorWare Products’ Properties as a value for the messages. With that, you can send SNMP Traps to the Windows internal SNMP Agent or any other device that is able to receive SNMP Traps. Of course you have full enterprise support, too. This gives you the possibility to involve every machine on your network into your security plan or whatever purpose it should serve.**

### Action Queue Options




*WinSyslog - Send SNMP Trap Action Queue Options*

### Use Diskqueue if connection to Syslog server fails

**File Configuration field:**

nUseDiscQueue

**Description:**

Enable diskqueuing syslog messages after unexpected connection loss.

### Split files if this size is reached

**File Configuration field:**

nDiskQueueMaxFileSize

**Description:**

Files are split when they reach this size in bytes. The default is 10485760 (10 MB). The maximum per file is 2147483648 bytes (2 GB). The total diskqueue size is limited by free disk space.

### Diskqueue Directory

**File Configuration field:**

szDiskQueueDirectory

**Description:**

The directory where the queue files will be generated in. The queuefiles will be generated with a dynamic UUID bound to the action configuration.

### Waittime between connection tries

**File Configuration fields:**

nDiskCacheWait

**Description:**

The minimum waittime until the Syslog Action retries to establish a connection to the Syslog server after failure.

### Overrun Prevention Delay (ms)

**File Configuration field:**

nPreventOverrunDelay

**Description:**

When the Action is processing syslog cache files, an overrun prevention delay can be added to avoid flooding the target Syslog server.

### Double wait time after each retry

**File Configuration field:**

bCacheWaittimeDoubling

**Description:**

If enabled, the configured waittime is doubled after each try.

### Limit wait time doubling to

**File Configuration field:**

nCacheWaittimeDoublingTimes

**Description:**

How many times the wait time is doubled after a failed connection try. This is a count, not seconds. The range is 1 to 100, and the default is 10.

### Enable random wait time delay

**File Configuration field:**

bCacheRandomDelay

**Description:**

If enabled, a some random time will be added into the waittime delay. When using many syslog senders, this can avoid that all senders start sending cached syslog data to the Syslog server at the same time.

### Maximum random delay

**File Configuration field:**

nCacheRandomDelayTime

**Description:**

Maximum random delay time that will be added to the configured waittime if Enable random wait time delay is enabled.

## Syslog Forwarding

### Protocol Type

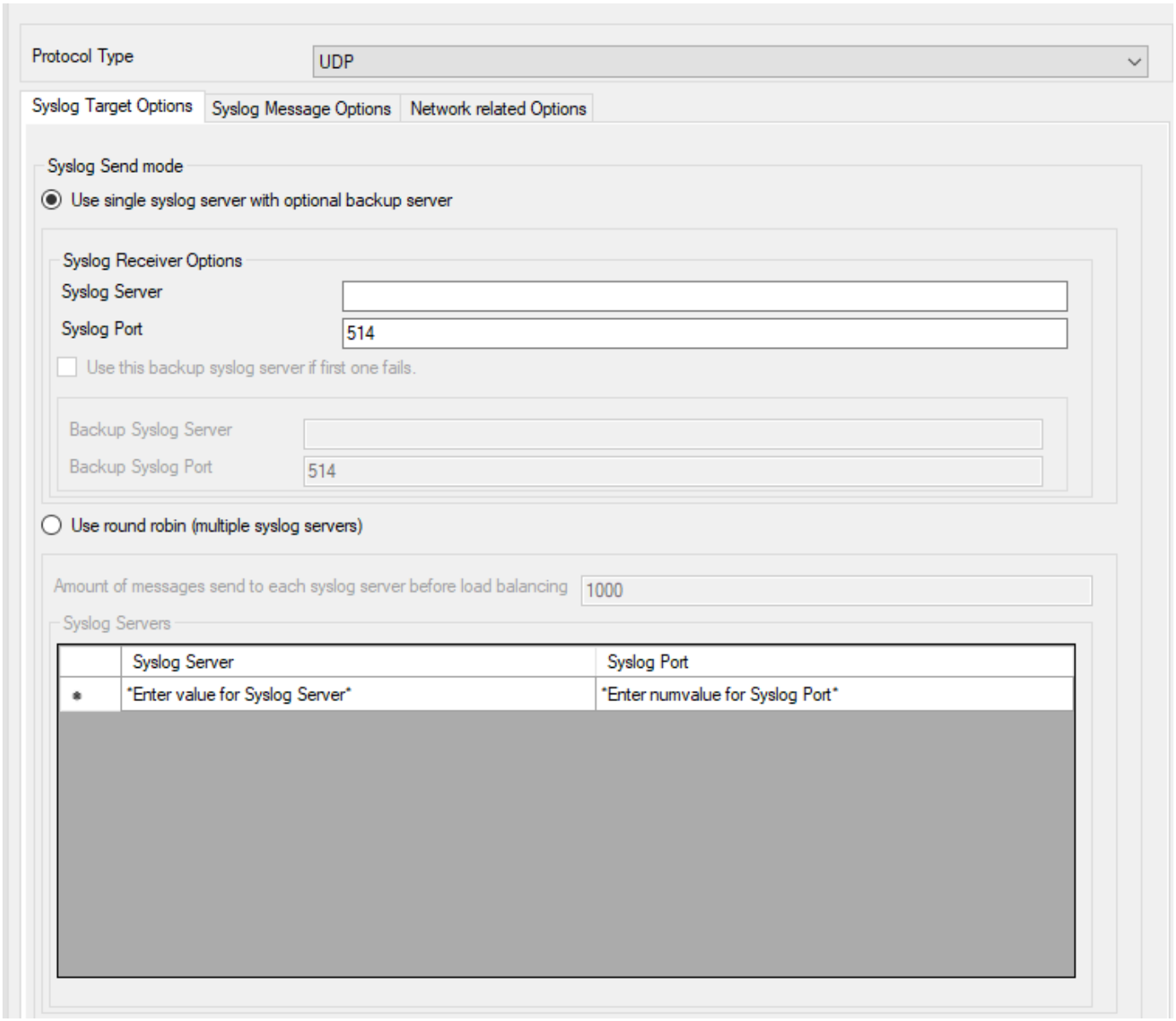
There are various ways to transmit syslog messages. In general,they can be sent via UDP, TCP, or RFC 3195 RAW. Typically, syslog messages are received via UDP protocol, which is the default. UDP is understood by almost all servers, but doesn't guarantee transport. In plain words, this means that syslog messages sent via UDP can get lost if there is a network error, the network is congested or a device (like a router or switch) is out of buffer space. Typically, UDP works quite well. However, it should not be used if the loss of a limited number of messages is not acceptable.

TCP and RFC 3195 based syslog messages offer much greater reliability. RFC 3195 is a special standardized transfer mode. However, it has not receive any importance in practice. Servers are hard to find. As one of the very few, Adiscon products support RFC 3195 also in the server implementations. Due to limited deployment, however, RFC 3195 is very little proven in practice. Thus we advise against using RFC 3195 mode if not strictly necessary (e.g. part of your requirement sheet).

TCP mode comes in three flavors. This stems back to the fact that transmission of syslog messages via plain TCP is not yet officially standardized (and it is doubtful if it ever will be). However, it is the most relevant and most widely implemented reliable transmission mode for syslog. It is a kind of unwritten industry standard. We support three different transmission modes offering the greatest compatibility with all existing implementations. The mode "TCP (one message per connection)" is a compatibility mode for Adiscon servers that are older than roughly June 2006. It may also be required for some other vendors. We recommend not to use this setting, except when needed. "TCP (persistent connection)" sends multiple messages over a single connection, which is held open for an extended period of time. This mode is compatible with almost all implementations and offers good performance. Some issues may occur if control characters are present in the syslog message, which typically should not happen. The mode "TCP (octet-count based framing)" implements algorithms of an IETF standard RFC 6587. It also uses a persistent connection. This mode is reliable and also deals with embedded control characters very well. This standard is now widely supported by modern syslog receivers and implementations.

As a rule of thumb, we recommend to use "TCP (octet-count based framing)" if you are dealing only with (newer) Adiscon products. Otherwise, "TCP (persistent connection)" is probably the best choice. If you select one of these options, you can also select a timeout. The connection is torn down if that timeout expires without a message being sent. We recommend to use the default of 30 minutes, which should be more than efficient. If an installation only occasionally sends messages, it could be useful to use a lower timeout value. This will free up connection slots on the server machine.

### Syslog Target Options




- Action - Forward Syslog Target Options*

### Syslog Send mode

**File Configuration field:**

nSendMode

**Description**

The Sendmode has been added since 2018 into all products supporting the forward syslog action. There are two options available.

**Use single Syslog server with optional backup server** This is the classic syslog send mode which uses a primary Syslog server and a secondary backup Syslog server if configured.

**Use round robin (multiple syslog servers)** This new method allows you to configure multiple targets that will be used one by one after a configured amount of messages has been sent to each target.

### Syslog server (Syslog Send mode)

**File Configuration field:**

szSyslogSendServer

**Description:**

This is the name or IP address of the system to which Syslog messages should be sent to. You can either use an IPv4, an IPv6 Address, or a Hostname that resolves to an IPv4 or IPv6 Address.

### Syslog Port (Syslog Send mode)

**File Configuration field:**

nSyslogSendPort

**Description:**

The remote port on the Syslog server to report to. If in doubt, please leave it at the default value of 514, which is typically the Syslog port. Different values are only required for special setups, for example in security sensitive areas. Set the port to 0 to use the system-supplied default value (which defaults to 514 on almost all systems).

Instead of the port number, a service name can be used. If so, that name is looked up via the socket service database functions.

### Use this backup Syslog server if first one fails

**File Configuration field:**

nEnableBackupServer

**Description:**

The backup server is automatically used if the connection to the primary server fails. The primary server is automatically retried when the next Syslog session is opened. This option is only available when using TCP syslog.

### Use round robin (multiple Syslog server)

### Amount of messages send to each Syslog server before load balancing

**File Configuration field:**

nRoundRobinMsgCount

**Description:**

When using round robin mode, this is the amount of messages to be sent to each configured Syslog server.

### Syslog server (Round robin mode)

**File Configuration field:**

szSyslogServer_[n]

**Description:**

This is the name or IP address of the system to which Syslog messages should be sent to. You can either use an IPv4, an IPv6 Address, or a Hostname that resolves to an IPv4 or IPv6 Address.

### Syslog Port (Round robin mode)

**File Configuration field:**

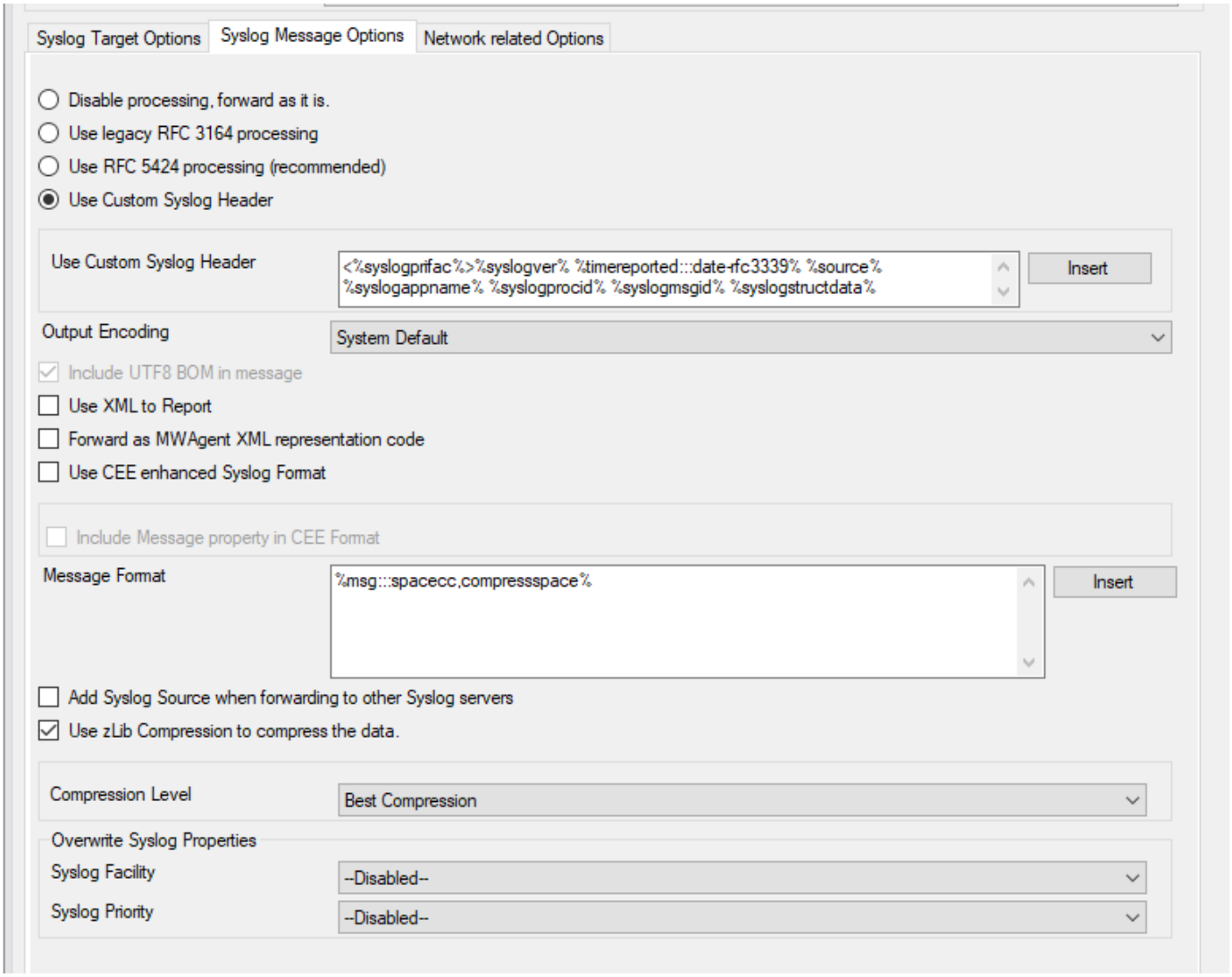
nSyslogPort_[n]

**Description:**

The remote port on the Syslog server to report to. If in doubt, please leave it at the default value of 514, which is typically the Syslog port. Different values are only required for special setups, for example in security sensitive areas. Set the port to 0 to use the system-supplied default value (which defaults to 514 on almost all systems).

Instead of the port number, a service name can be used. If so, that name is looked up via the socket service database functions.

### Syslog Message Options




- Action - Forward Syslog - Message Options*

### Syslog processing

**File Configuration field:**

bProcessDuringRelay

- 0 = Disable processing, forward as it is
- 1 = RFC3164 Header - Use legacy RFC 3164 processing
- 2 = RFC5424 Header - Use RFC 5424 processing (recommended)
- 3 = Custom Syslog Header

**Description:**

With this settings you can assign how your syslog messages will be processed.

For processing syslog you can choose out of four different options. You can use rfc3164 or RFC5424 (recommended) which is the current syslog standard, you are able to customize the syslog header or you do not process your syslog and forwards it as it is.

### Use Custom Syslog Header

**File Configuration field:**

szCustomSyslogHeader

**Description:**

In this field you can specify the contents of your syslog header. This option is only available when you choose “Use Custom Syslog Header” in the Syslog Processing menu. The contents can be either a fixed message part which you can write into the field yourself or you use properties as dynamic content. By default the Header field is filled with the content of the RFC 5424 header.

**Please note** that the header content of the Header field can be configured. event properties are described in the property replacer section.

### Output Encoding

**File Configuration field:**

nOutputEncoding

**Description:**

This setting is most important for Asian languages. A good rule is to leave it at “System Default” unless you definitely know you need a separate encoding. “System Default” works perfect in the far majority of cases, even on Asian (e.g. Japanese) Windows versions.

### Include UTF8 BOM in message

**File Configuration field:**

nProtocolType

**Description:**

If enabled (default), the UTF8 BOM code will be prepended to the output message if you are using UTF8 Output encoding. If the syslog receiver cannot handle and remove the UTF8 BOM you can disabled this option.

### Use XML to Report

**File Configuration field:**

bReportInXML

**Description:**

If this option is checked, the forwarded Syslog message is a complete XML-formatted information record. It includes additional information like timestamps or originating system in an easy to parse format.

The XML formatted message is especially useful if the receiving system is capable of parsing XML data. However, it might also be useful to a human reader as it includes additional information that cannot be transferred otherwise.

### Forward as WinSyslog XML Representation Code

**File Configuration field:**

nForwardIUT

**Description:**

MonitorWare supports a specific XML-Representation of the event. If it is checked, that XML representation is used. It provides additional information (like informationunit type, original source system, reception time & many more) but is harder to read by a human. At the same time, it is obviously easier to parse.

### Use CEE enhanced Syslog Format

**File Configuration field:**

nReportInJSON

**Description:**

If enabled, the CEE enhanced Syslog format will be used. All useful properties will be included in a JSON Stream. The message itself can be included as well, see the “Include message property in CEE Format” option. Here is a sample how the format looks like for a security Eventlog message:

```
@cee: {"source": "machine.local", "nteventlogtype": "Security", "sourceproc": "Microsoft-Wi
ndows-Security-Auditing", "id": "4648", "categoryid": "12544", "category": "12544", "keyw
```

```
ordid": "0x8020000000000000", "user": "N\A", "SubjectUserSid": "S-1-5-11-222222222-33333333
3-4444444444-5555", "SubjectUserName": "User", "SubjectDomainName": "DOMAIN", "SubjectLogon
Id": "0x5efdd", "LogonGuid": "{00000000-0000-0000-0000-000000000000}", "TargetUserName": "A
dministrator", "TargetDomainName": " DOMAIN ", "TargetLogonGuid": "{00000000-0000-0000-0000
-000000000000}", "TargetServerName": "servername", "TargetInfo": " servername ", "ProcessId
": "0x76c", "ProcessName": "C:\Windows\System32\spoolsv.exe", "IpAddress": "-", "IpPort": "
-", "catname": "Logon", "keyword": "Audit Success", "level": "Information", }
```


Additionally to this format you can set: Include message property in CEE Format.

If enabled, the message itself will be included in the JSON Stream as property. Disable this option if you do not want the message itself in the CEE Format.

If you want the Event ID to appear directly inside the forwarded syslog message text, use the custom message format in the Forward Syslog action and insert the Event ID property explicitly. This is useful when the receiving system primarily parses the message text instead of structured properties.

### Include message property in CEE Format

**Description**

If enabled, the message itself will be included in the JSON Stream as property. Disable this option if you do not want the message itself in the CEE Format.

### Message Format

**File Configuration field:**

szMessageFormat

**Description:**

The custom format lets you decide how the content of a syslog message looks like. You can use properties to insert content dynamically or have fixed messages that appear in every message. Event properties are described in the property replacer section.

### Add Syslog Source when forwarding to other Syslog servers

**File Configuration field:**

nSyslogInsertSource

**Description:**

If this box is checked, information on the original originating system is prepended to the actual message text. This allows the recipient to track where the message originally came from.

**Please note:** This option is not compatible with RFC 3164. We recommend selecting it primarily when message forwarding to a WinSyslog Interactive Server is intended.

### Use zLib Compression to compress the data

**File Configuration field:**

nUseCompression

**Description:**

With this option you can set the grade of compression for your syslog messages. For more information please read the note at the bottom of this page.

### Compression Level

**File Configuration field:**

nCompressionLevel

- 1 = Best Speed
- 3 = Low Compression
- 6 = Normal Compression

- 9 = Best Compression (default)

**Description:**

With this option you can set the grade of compression for your syslog messages. For more information please read the note at the bottom of this page.

**Note on Using Syslog Compression**

Compressing syslog messages is a stable but rarely used feature. There is only a very limited set of receivers who are able to understand that format. Turning on compression can save valuable bandwidth in low-bandwidth environments. Depending on the message, the saving can be anything from no saving at all to about a reduction in half. The best savings ratios have been seen with Windows Event Log records in XML format. In this case, 50% or even a bit more can be saved. Very small messages do not compress at all. Typical syslog traffic in non-xml format is expected to compress around 10 to 25%.

Please note that compression over TCP connections requires a special transfer mode. This mode uses OpenSSL TLS Implementation 3.x for secure transmission. TLS compression is not implemented; instead, the system uses standard OpenSSL compression mechanisms.

Besides the fact that the mechanisms behind compression are experimental, the feature itself is solid.

### Overwrite Syslog Properties

### Syslog Facility

**File Configuration field:**

nSyslogFacility

**Description:**

When configured, will overwrite the Syslog Facility with the configured value.

### Syslog Priority

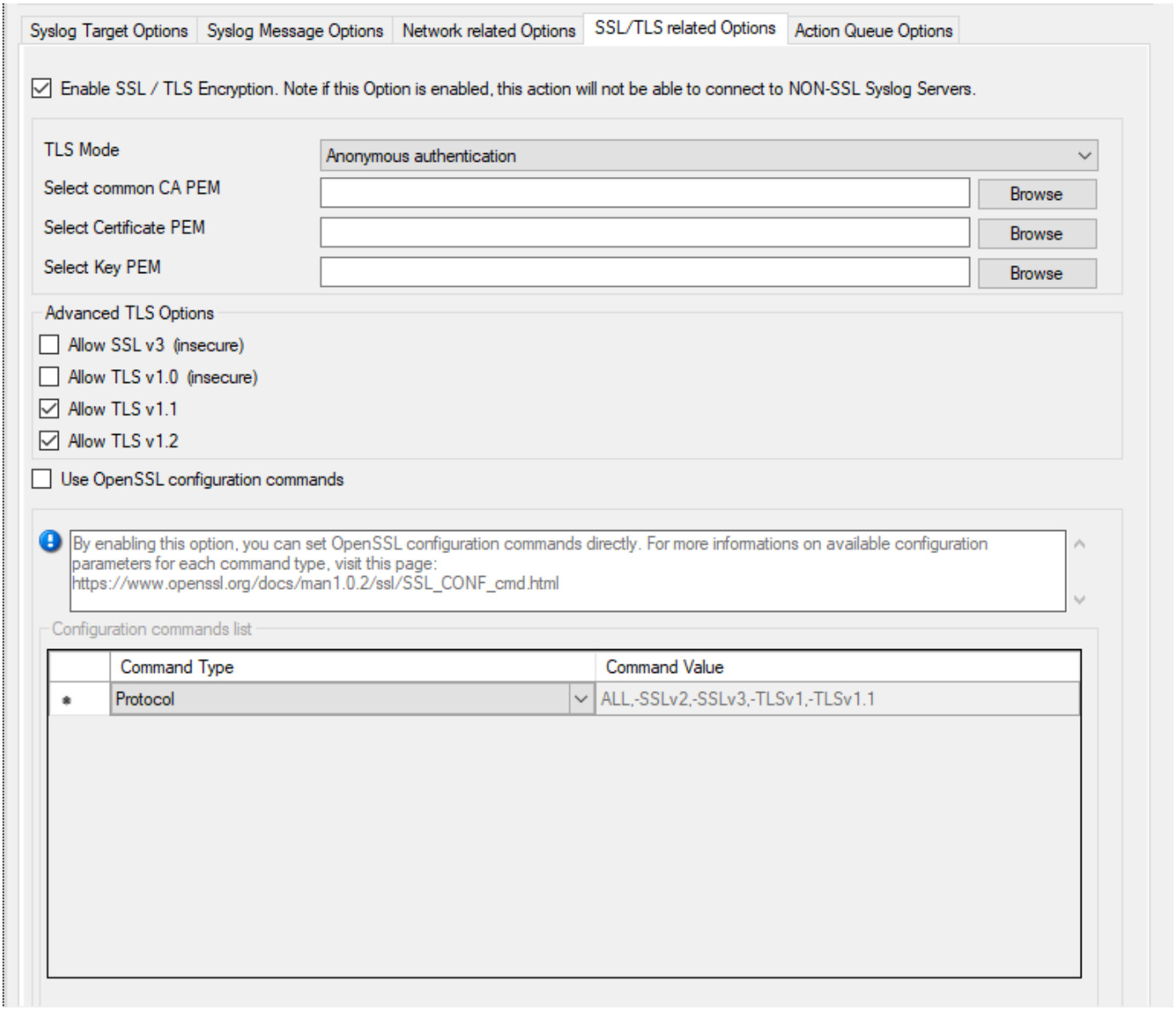
**File Configuration field:**

nSyslogPriority

**Description:**

When configured, will overwrite the Syslog Priority with the configured value.

### SSL/TLS related Options




- Action - Forward Syslog SSL/TLS related Options*

### Enable SSL / TLS Encryption

**File Configuration field:**

nUseSSL

**Description:**

If this option is enabled, the action will not be able to talk to a NON-SSL secured server. The method used for encryption is compatible to RFC5425 (Transport Layer Security (TLS) Transport Mapping for Syslog).

### TLS Mode

**File Configuration field:**

nTLSMode

**Description:**

**Anonymous Authentication**

Default option. This means that a default certificate will be used.

**Use Certificate**

If this option is enable, you can specify your own certificate. For further authentication solutions, you will need to create your own certificates using OpenSSL Tools for example.

### Select common CA PEM

**File Configuration field:**

szTLSCAFile

**Description:**

Select the certificate from the common Certificate Authority (CA). The syslog receiver should use the same CA.

### Select Certificate PEM

**File Configuration field:**

szTLSCertFile

**Description:**

Select the client certificate (PEM Format).

### Select Key PEM

**File Configuration field:**

szTLSKeyFile

**Description:**

Select the keyfile for the client certificate (PEM Format).

### Allow SSL v3

**File Configuration field:**

nTLSAllowSSLv3

**Description:**

This option enables insecure protocol method SSLv3. We recommend NOT enabling this option as SSLv3 is considered broken.

### Allow SSL v1.0

**File Configuration field:**

nTLSAllowTLS10

**Description:**

This option enables insecure protocol method TLSv1. We recommend NOT enabling this option as TLSv1 is considered broken.

### Allow SSL v1.1

**File Configuration field:**

nTLSAllowTLS11

**Description:**

This option enables protocol method TLS1.1 which is enabled by default.

### Allow SSL v1.2

**File Configuration field:**

nTLSAllowTLS12

**Description:**

This option enables protocol method TLS1.2 which is enabled by default.

### Allow TLS v1.3

**File Configuration field:**

nTLSAllowTLS13

**Description:**

This option enables protocol method TLS1.3 which provides enhanced security and performance.

### Use OpenSSL configuration commands

**File Configuration field:**

nTLSUseConfigurationCommands

**Description:**

By enabling this option, you can set OpenSSL configuration commands directly. For more information's on available configuration parameters for each command type, visit this page:

https://www.openssl.org/docs/man1.0.2/ssl/SSL_CONF_cmd.html

We allow to the set the following OpenSSL configuration commands in the configuration commands list:

- CipherString: Sets the allowed/disallowed used Ciphers. Setting this value will OVERWRITE the internal default ciphers.
- SignatureAlgorithms: This sets the supported signature algorithms for TLS v1.2.
- Curves: This sets the supported elliptic curves.
- Protocol: Sets the supported versions of the SSL or TLS protocol. This will OVERWRITE the Allow SSL options from above!
- Options: The value argument is a comma separated list of various flags to set.

When setting advanced configuration commands, we highly recommend to enable debug logging and review it after changes have been made. An error will be logged in the debug logfile if a configuration command cannot be processed successfully.

### TCP related Options

When using TCP-based syslog forwarding, you have the additional option to use the diskqueue. Whenever a connection to a remote Syslog server fails, the action starts caching the syslog messages into temporary files. The folder for these files can be configured. The filenames are generated using a unique GUID which is automatically generated for each Action, thus enabling you to use this feature in multiple Actions. Once the Syslog server becomes available again, the cached messages are being sent automatically. If you restart the Service while the Syslog Cache was active, it cannot be checked during service startup if the Syslog server is available now. Once the action is called again, the check is done and if the Syslog server is available, the messages are being sent. The size of this cache is only limited by the disk size. Files are split by 10MB by default, but this can also be configured. The maximum supported file size is 2GB.

**Please Note:** This option is not available for UDP or RFC 3195.

### Session Timeout

**File Configuration field:**

nTimeoutValue

**Description:**

Controls when an idle TCP session is closed, including persistent and octet-count based framing connections. This is separate from the TLS Send Progress Timeout, which limits a TLS operation that stops making progress.

### TLS Send Progress Timeout

**File Configuration field:**

nTLSSendTimeout

**Availability:**

Scheduled for version 26.11. The setting and the timeout and cancellation guarantees below require service version 26.11 or later. Updating the configuration client alone does not add this capability to an older service.

**Description:**

Maximum time without successful TLS send progress, in milliseconds. The default is 30000 (30 seconds). Valid values are 1 through 2147483647. Missing or invalid values use the default; zero does not disable the timeout.

This setting applies when forwarding Syslog over TCP with TLS enabled, including persistent connections, reconnects, backup destinations, and diskqueue replay. It does not apply to UDP or unencrypted TCP forwarding.

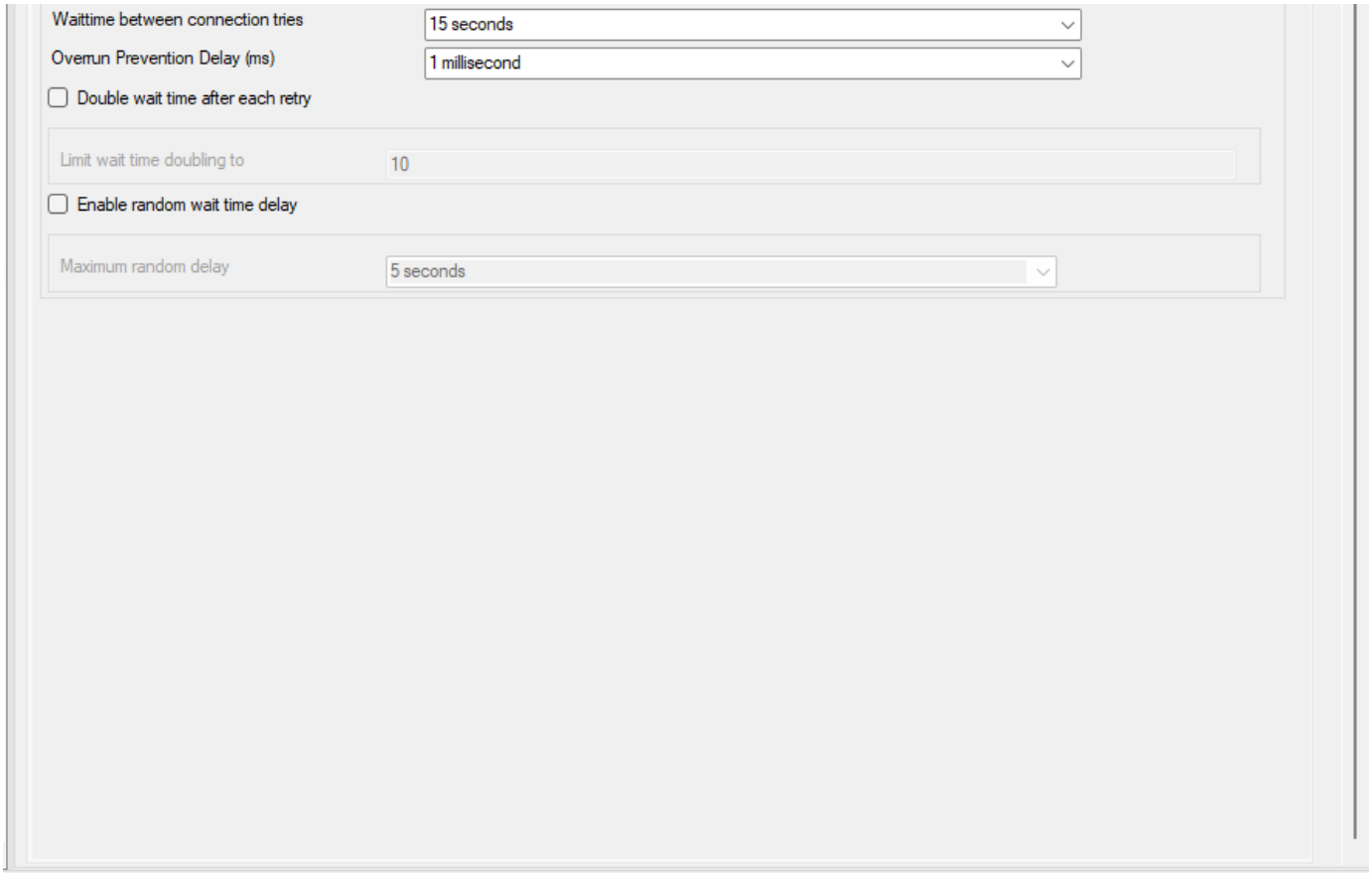
Each successful TLS write, including a partial write, restarts the inactivity interval. Socket readiness and TLS retry requests do not restart it. A send can therefore take longer than the configured interval while it continues making progress. Successful writes do not confirm that the destination has processed the message.

TCP connection establishment and the TLS handshake share one total deadline with the same configured value. Progress during these steps does not restart that deadline. Name resolution is also bounded by the configured value, using a separate deadline before the connection attempt.

On timeout, the failed connection is closed and the action uses its existing transmission-failure handling. If diskqueuing is enabled and persistence succeeds, unsent messages are retained for later replay. A timeout does not immediately trigger a reconnect or a backup attempt. Partially transmitted messages may be duplicated when replayed.

Service stop and configuration reload cancel an active TLS network operation without waiting for this timeout. The Session Timeout setting (`nTimeoutValue`) instead controls closure of an idle TCP session; it does not bound an active TLS send.

### Action Queue Options




*WinSyslog - Forward Syslog Action Queue Options*

### Use Diskqueue if connection to Syslog server fails

**File Configuration field:**

nUseDiscQueue

**Description:**

Enable diskqueuing syslog messages after unexpected connection loss.

### Split files if this size is reached

**File Configuration field:**

nDiskQueueMaxFileSize

**Description:**

Files are split when they reach this size in bytes. The default is 10485760 (10 MB). The maximum per file is 2147483648 bytes (2 GB). The total diskqueue size is limited by free disk space.

### Diskqueue Directory

**File Configuration field:**

szDiskQueueDirectory

**Description:**

The directory where the queue files will be generated in. The queuefiles will be generated with a dynamic UUID bound to the action configuration.

### Waittime between connection tries

**File Configuration fields:**

nDiskCacheWait

**Description:**

The minimum waittime until the Syslog Action retries to establish a connection to the Syslog server after failure.

### Overrun Prevention Delay (ms)

**File Configuration field:**

nPreventOverrunDelay

**Description:**

When the Action is processing syslog cache files, an overrun prevention delay can be added to avoid flooding the target Syslog server.

### Double wait time after each retry

**File Configuration field:**

bCacheWaittimeDoubling

**Description:**

If enabled, the configured waittime is doubled after each try.

### Limit wait time doubling to

**File Configuration field:**

nCacheWaittimeDoublingTimes

**Description:**

How many times the wait time is doubled after a failed connection try. This is a count, not seconds. The range is 1 to 100, and the default is 10.

### Enable random wait time delay

**File Configuration field:**

bCacheRandomDelay

**Description:**

If enabled, a some random time will be added into the waittime delay. When using many syslog senders, this can avoid that all senders start sending cached syslog data to the Syslog server at the same time.

### Maximum random delay

**File Configuration field:**

nCacheRandomDelayTime

**Description:**

Maximum random delay time that will be added to the configured waittime if Enable random wait time delay is enabled.

### UDP related Options

### Enable IP Spoofing for the UDP Protocol

**File Configuration field:**

nSpoofIPAddress

**Description:** This option enables you to spoof the IP Address when sending Syslog messages over UDP. Some notes regarding the support of IP Spoofing. It is only supported the UDP Protocol and IPv4. IPv6 is not possible yet. Due to system limitations introduced by Microsoft, IP spoofing support is limited and may not be available on Windows client editions (for example Windows 10/11). Also please note that most routers and gateways may drop network packages with spoofed IP Addresses, so it may only work in local networks.

### Fixed IP or single property

**File Configuration field:**

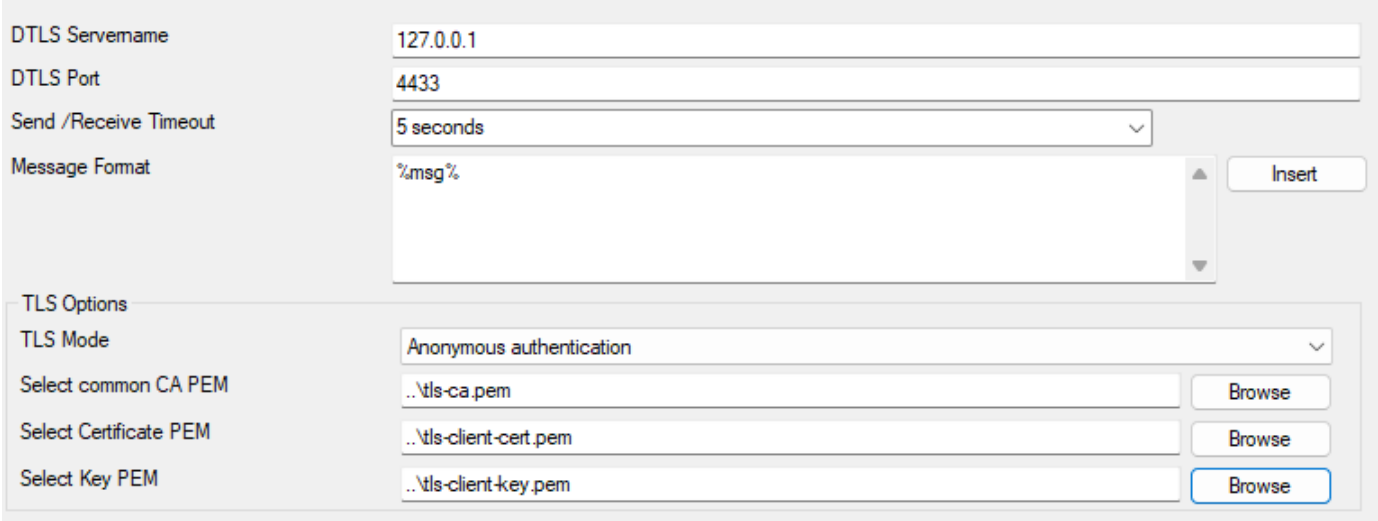
szSpoofedIPAddress

**Description:**

You can either use a static IP Address or a property. When using a property, the IP Address is tried to be resolved from the content of the property. For example by default the `%source%` property is used. If the name in this property cannot be resolved to an IP Address, the default local IP Address will be used.

### Send DTLS

This action sends messages securely using Datagram Transport Layer Security (DTLS). Use it when the receiving system expects DTLS transport for forwarded messages.




- Action - Send DTLS Configuration*

### DTLS Servername

**File Configuration field:**

szDTLSServer

**Description:**

The hostname or IP address of the DTLS server to which messages should be sent. You can use an IPv4, IPv6 address, or a hostname resolving to one of these.

### DTLS Port

**File Configuration field:**

nDTLSSendPort

**Description:**

The port number on the DTLS server where messages are sent. Typically, this port is 4433.

### Send/Receive Timeout

**File Configuration field:**

nSendTimeOut

**Description:**

Specifies the time in seconds to wait for a response from the DTLS server before timing out. For instance, set this value to “5 seconds” for a 5-second timeout.

### Message Format

**File Configuration field:**

szMessage

**Description:**

Defines the format of the message being sent. Use placeholders like “%msg%” to define the dynamic content of the message. Multi-line messages are supported.

### TLS Options

#### TLS Mode

**File Configuration field:**

nTLSMode

**Description:**

Specifies the authentication method used. Options include “Anonymous authentication” or other modes requiring certificates.

#### Select common CA PEM

**File Configuration field:**

szTLSCAFile

**Description:**

Path to the CA certificate file (e.g., *tls-ca.pem*).

#### Select Certificate PEM

**File Configuration field:**

szTLSCertFile

**Description:**

Path to the client certificate file (e.g., *tls-client-cert.pem*).

#### Select Key PEM

**File Configuration field:**

szTLSKeyFile

**Description:**

Path to the private key file for the client (e.g., *tls-client-key.pem*).

### HTTP Request

The **HTTP Request** action sends an HTTP or HTTPS request to a remote host. It is listed under **Forwarding** in the configuration client. Use it for simple HTTP callbacks where the **HTTP REST Output** action is not required.




*Action - HTTP Request*

#### Host

**File Configuration field:**

szHost

**Description:**

Specify the targeted host here.

#### URL & Querystring

**File Configuration field:**

szUrl

**Description:**

By default this is `/index.html`. This value is used to construct an URL which is previewed in a rectangular field under Use secure https Protocol option.

#### Port

**File Configuration field:**

nPort

**Description:**

This port is to be probed. Please see your server's reference for the actual value to use. For example, mail servers typically listen to port 25 and web servers to 80.

#### Referrer

**File Configuration field:**

szReferer

**Description:**

An optional configuration option where you can specify a Referrer that is send in the HTTP header.

#### UserAgent (Browser)

**File Configuration field:**

szUserAgent

**Description:**

It is also an optional value which can be used to specify a UserAgent that is send in the HTTP header.

## Internal processing actions

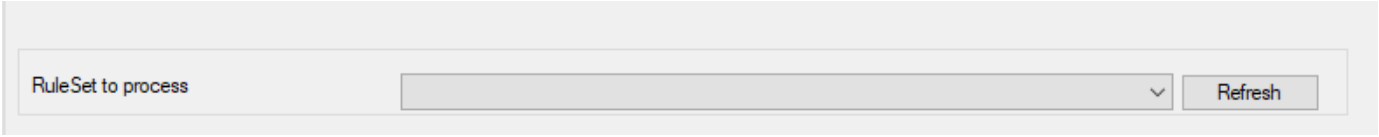
### Call RuleSet

A Call RuleSet action simply calls another ruleset in some existing ruleset. When this action is encountered, the rule engine leaves the normal flow and goes to the called ruleset (which may contain many rules as well). It executes all the rules that have been defined in the called RuleSet. After the execution of all of them, it will return to its point from where it left the original flow. Let's take an example to clarify it a little further.

Let's say that Rule 1 has two actions - Action 1 and Action 2. The Action 1 of Rule 1 is an include (Call Ruleset) action. If the filter condition result of Rule 1 evaluates to true, it will execute the Action 1. Since Action 1 is the include action in this example, it will go to the included ruleset and will execute its filter condition. If that filter condition evaluates to true, it will execute all of its actions and will return to Action 2 of Rule 1 (of normal flow). If on the other hand, the filter condition of the included rule set evaluates to false, it will skip all of its actions and will come back to the Action 2 of Rule 1 (of normal flow).

**Note:** there is no limit on including the rules which means that a rule

that has been included in another rule may contain another rule in it which might contain another rule in it and so on.




- Action - Call RuleSet*

### Ruleset to Call

**File Configuration field:**

szRuleSet

**Description:**

Select the Ruleset to be called.

**Note:** Call RuleSet stays disabled until you have more then "One" RuleSet!

### Compute Status Variable

An internal action used to compute a status variable. This is needed for RuleSets which operate on a counter basis. This dialog controls the compute status variable options.




- Action - Compute Status Variable*

#### Status variable

**File Configuration field:**

szStatusVar

**Description:**

Name of the unique status variable.

**Operation Type**

#### Increment Value

**File Configuration field:**

nCalcType = 1

**Description:**

It increments the value by the operation value.

#### Decrement Value

**File Configuration field:**

nCalcType = 2

**Description:**

It decrements the value by the operation value.

#### Operation value

**File Configuration field:**

nChangeVal

**Description:**

The operation value that is to be used.

### Discard

A Discard Action immediately destroys the current Information Unit and any action of any rule that has been defined after the Discard action execution. When this action has been selected then no dialog appears as nothing needs to be configured for this.

Use the Discard action when matching events should be ignored and no later actions in the same rule chain should run for them.

### Normalize Event

Parameters can be normalized and converted into XML, CSV, and JSON formats. The normalization result is stored into an internal property which can be used for filtering decisions as well as for output actions.

The action uses liblognorm (http://www.liblognorm.com/files/manual/index.html) which is also used by rsyslog. Rulebases created for liblognorm can easily be used and adapted.




- Action - Normalize Event*

### Normalize Parameter

**File Configuration field:**

szMessage

**Description:**

Specifies the property that you want to normalize, by default this is the `%msg%` property.

### Select Rulebase File

**File Configuration field:**

szRulebase

**Description:**

The text file that contains the rulebase definitions (see liblognorm documentation for more).

### Lognorm Output Format

**File Configuration field:**

nOutputFormat

- 0 = DISABLED
- 1 = JSON
- 2 = XML
- 3 = CSV

**Description:**

- Disabled: No additional output format.
- JSON Format: Creates a string formatted in JSON which is stored in the output property.
- XLM Format: Creates an XML formatted string which is stored in the output property.
- CSV Format: Creates a CSV (Comma separated values) string which is stored in the output property.

### Output Property

**File Configuration field:**

szOutputProperty

**Description:**

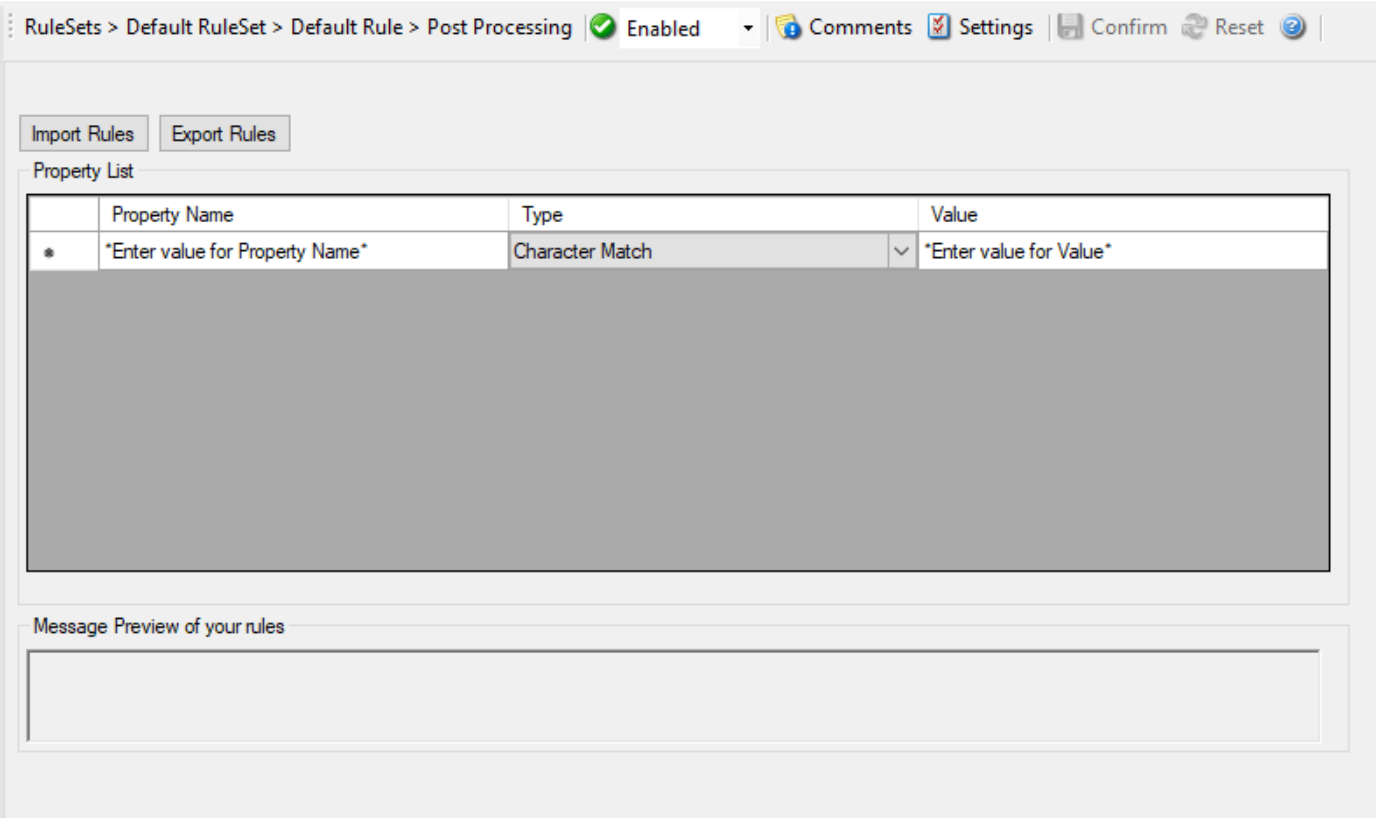
The property where the normalized format is saved to.

### Post Processing

The Post Processing action allows you to re-parse a message after it has been processed e.g. Tab Delimited format.

Such re-parsing is useful if you either have a non-standard Syslog format or if you would like to extract specific properties from the message.

The post process action takes the received message and parses it according to a parse map. The parse map specifies which properties of which type are present at which position in the message. If the message actually matches the parse map, all properties are extracted and are set as part of the event. If the parse map does not match the message, parsing stops at the first-non matching entry.




- Action - Post Processing*

### Templates

Parse maps can be quite complex. In order to facilitate exchange for parse maps, they can be persisted to XML files. Adiscon also plans to provide parse maps for some common devices.

We know that creating a parse map is often not a trivial task. If you are in doubt how to proceed, please contact us via the Customer Service System - we will happily assist you with your needs. In this case, you will probably receive a parse map file that you can import here.

### The Parse Map Editor

In this dialog, you can edit only in the text boxes above the data grid. When you select an entry in the grid, its values are updated in the textboxes. Any edits made there will automatically be reflected to the grid. Pressing Insert or Delete will create a new entry or delete the currently selected one.

### Property Name

**File Configuration field:**

szProperty_[n]

**Description:**

The property name that is to be parsed. The list box is pre-populated with standard and event properties. However, you can add any property name you like. If you create your own properties, we highly recommend

prefixing their name with “u-” so that there will be no duplicates with standard properties. Adiscon will never prefix any properties with “u-”. For example, if you would like to create a custom property “MyProperty”, we highly suggest that you use the property name “u-MyProperty” instead.

The property name “Filler” is reserved. Any values assigned to the Filler-property will be discarded. This is the way to get rid of fill-characters that you do not really need.

### Type

**File Configuration field:**

nSyntax_[n]

- Integer = 101
- IPV4Addr = 102
- CharMatch = 201
- RestOfMessage = 202
- Word = 203
- UpTo = 204
- TimeStampISO = 301
- TimeStampUNIX = 302

**Description**

Some types need an additional value. If that is needed, you can provide it here.

### Value

**File Configuration field:**

szParsValue_[n]

**Description:**

Some types need an additional value. If that is needed, you can provide it here.

### Message Preview

This is a read-only box. It shows a hypothetical message that would match the configured parsing rules.

## Parsing log messages

**This article describes how to parse log message via “Post-Process”. It illustrates the logic behind Post-Process action.**

### Get relevant information from logs

Log files contain a lot of information. In most cases only a small part of the log message is of actual interest. Extracting relevant information is often difficulty. Due to a variety of different log formats a generic parser covering all formats is not available.

Good examples are firewalls. Cisco PIX and FortiGate firewalls both use syslog for logging. But the content of their respective log messages are very different. Therefore a method is needed to parse the logs in a generic way. Here Post-Process action of Adiscon’s MonitorWare comes into play.

### Tool kit for parsing

Post-Process action provides an editor for creating a log format template. A template consists of as many rules as necessary to parse out the relevant information.

### Determine necessary information

In order to parse out information it is vital to know the exact structure of the message. Identifying the position of each relevant item is essential. Assuming for auditing purposes the following items are needed:

- Timestamp
- Source IP-Address
- SyslogTag
- MessageID
- Username
- Status
- Additional Information

A sample message looks like:

```
Mar 29 08:30:00 172.16.0.1 %Access-User: 12345: rule=monitor-user-login user=Bob status=denied msg=User does not exist
```

In order to extract the information let us examine each item within the message. Splitting the message makes it easier to explain. So here we go.

```
Pos = Position of the character
*p  = Points to the position the parser stands after parsing the*
      rule
Log = Message subdivided into its characters.
Pro = Property. In the term of Adiscon a property is the name of
      the item which is parsed out.mk:
      @MSITStore:C:\PROGRA~2\MONITO~1\Agent\manual\MONITO~1.CHM::/
```

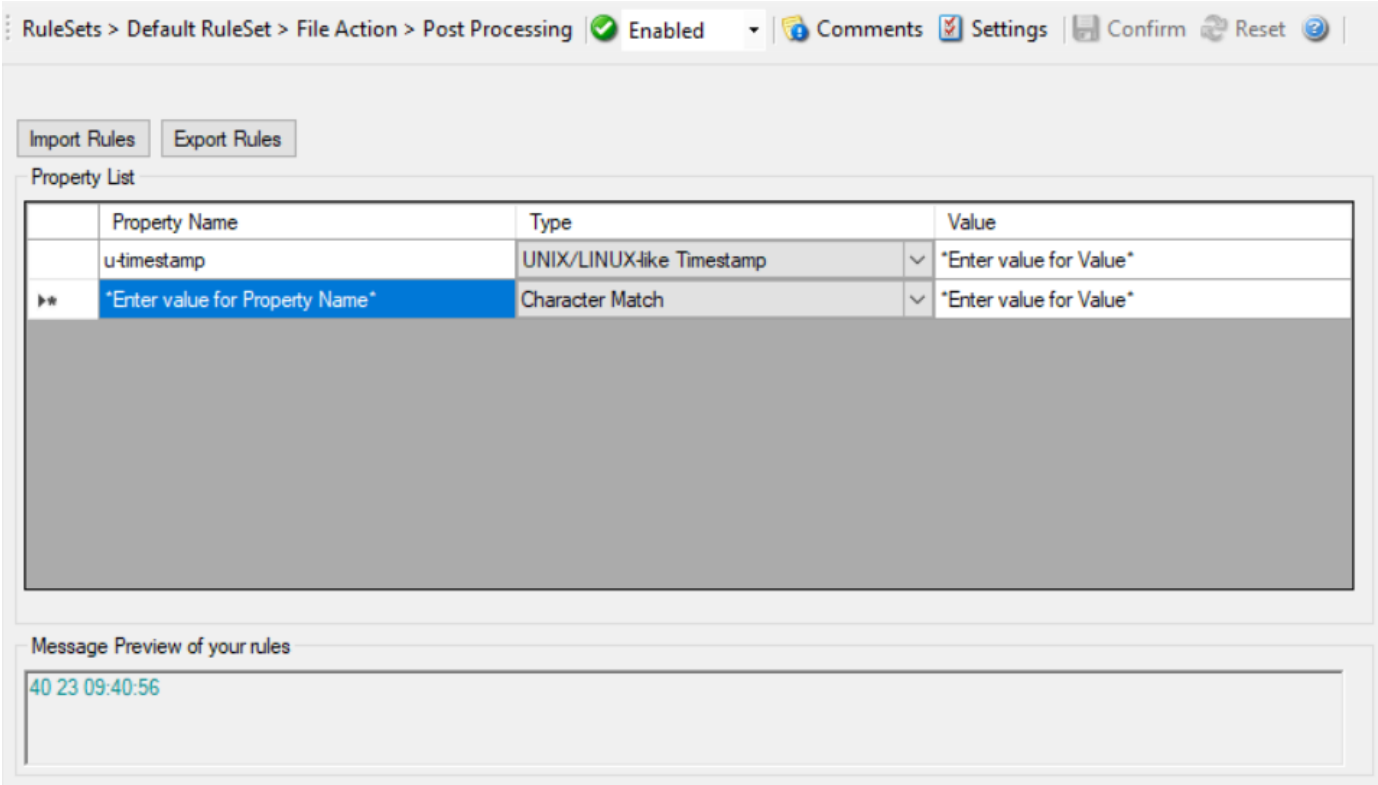
| Pos | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Log | M | a | r |  | 2 | 9 |  | 0 | 8 | : | 3 | 0 | : | 0 | 0 |  | 1 | 7 | 2 | . |
| *p | * |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Pro |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Note that at beginning of the parse process the parser’s pointer points to the first character. Each parse type starts parsing at the current position of the pointer.

### Parsing out a Timestamp

The first identified item is a so called Unix/Timestamp. It has always a length of 15 characters. ‘UNIX/LINUX-like Timestamp’ parse type exactly covers the requirement to parse this item. Therefore insert a rule and select ‘UNIX/LINUX-like Timestamp’ type. This rule parses out the timestamp and moves the pointer to the next character after the timestamp. Name the property ‘u-timestamp’ [1].

**Note:** There is a second timestamp-type, the **ISO-like-timestamp**. It has the format**2006-07-24 13:37:00.**



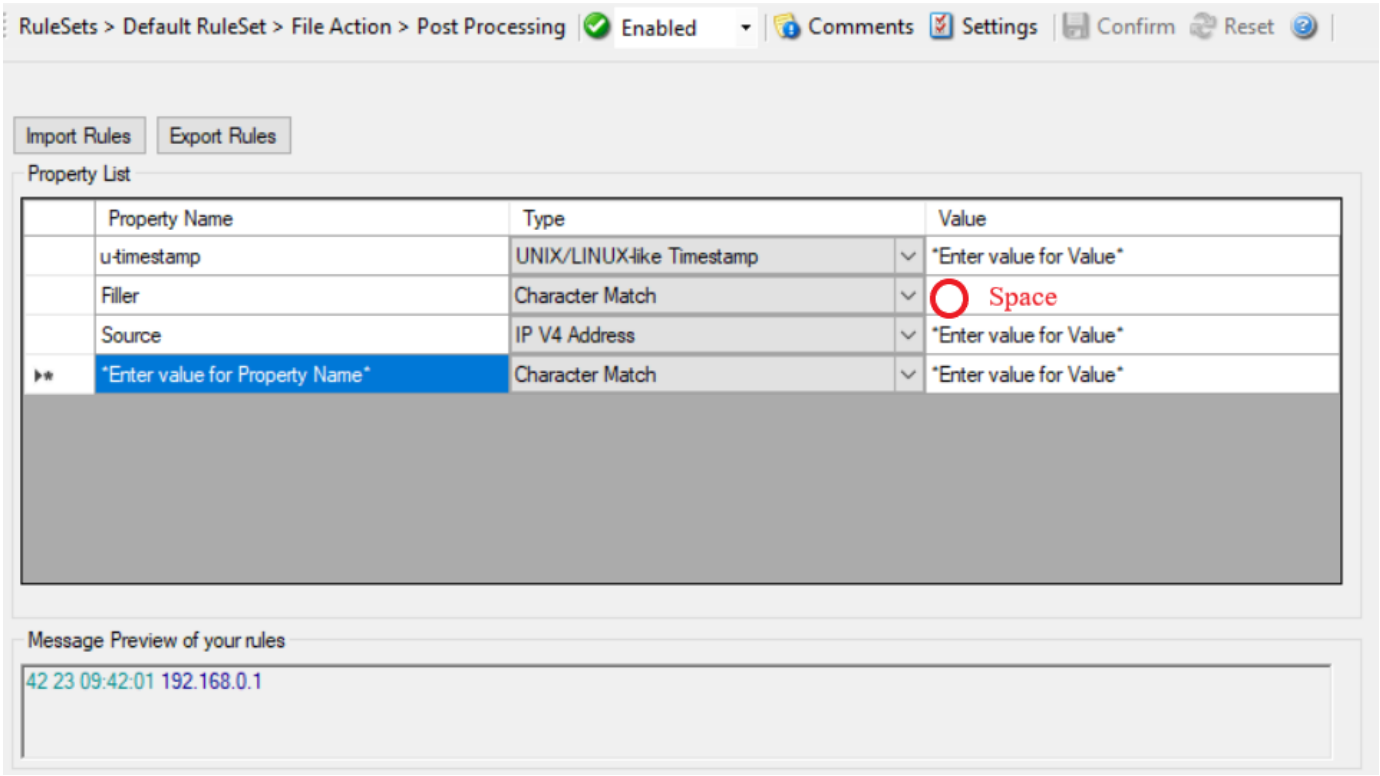

- Post-Process Editor: Inserted a ‘UNIX/LINUX like timestamp’ rule*

| Pos | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Log | M | a | r | | 2 | 9 | | 0 | 8 | : | 3 | 0 | : | 0 | 0 | | 1 | 7 | 2 | . |
| *p | | | | | | | | | | | | | | | | * | | | | |
| Pro | u-timestamp | | | | | | | | | | | | | | | | | | | |

### Get the IP-Address

Next item is the IP address. Note that after the timestamp follows a space and then the IP address. Therefore insert a ‘Character Match’ rule with a space as value. Select the ‘Filler’ [2] property for this rule. ‘Character Match’ requires a user defined value. This parse type compares the given value with the character at the current position of the message. The character has to be identical with the given value otherwise the parse process will fail. After applying this parse type the parse pointer is moved to the position immediately after the given value. In our sample this is the start position of the IP Address (position 17).

After that the address can be obtained. Place in a ‘IP V4 Address’ type. This type parses out a valid IP regardless of its length. No need to take care about the characters. Select ‘Source’ property or name it to whatever you prefer. The parser will automatically move the pointer to the position next to the address.



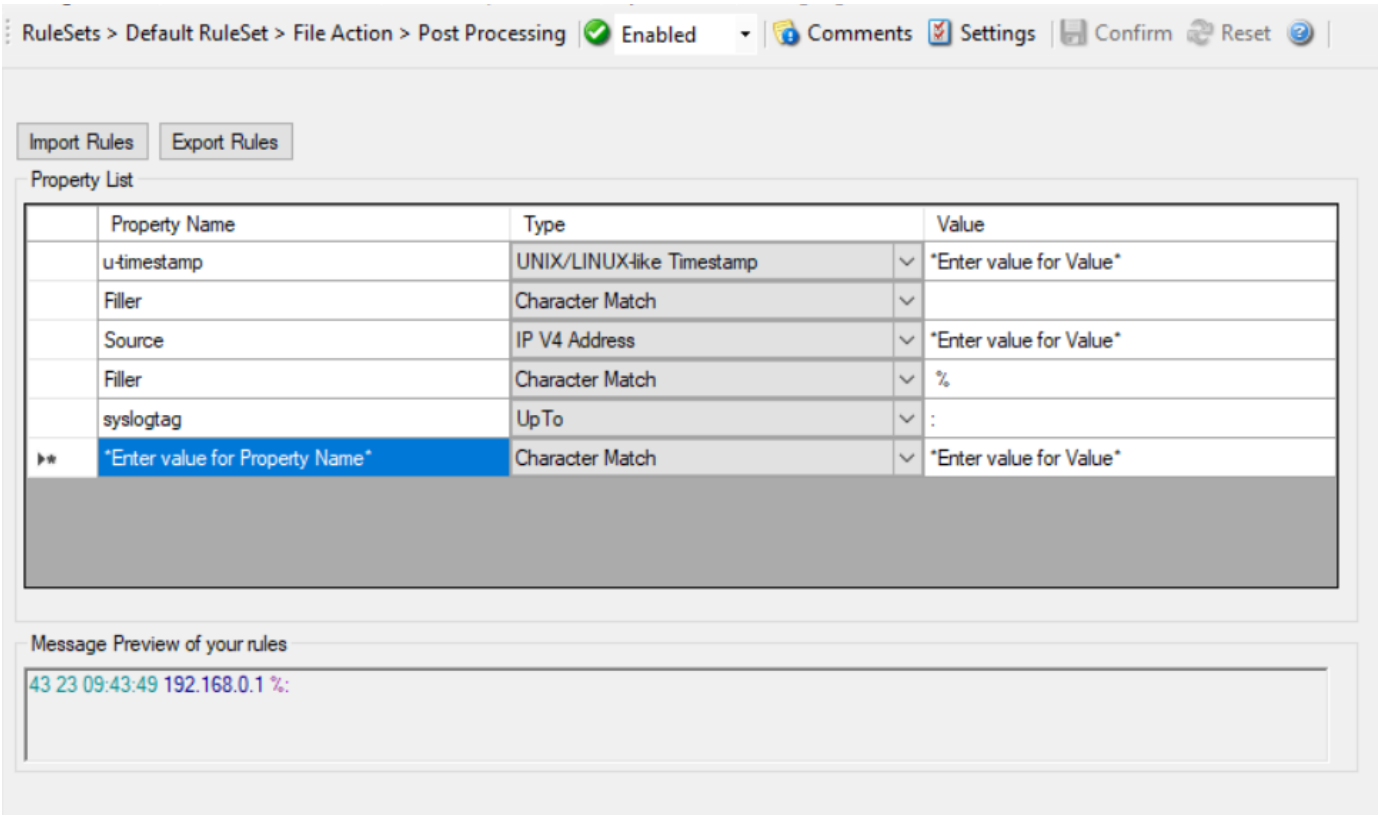

- Post-Process Editor: Note the value of ‘Character Match’ rule is a Space*

| Pos | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Log | 0 | | 1 | 7 | 2 | . | 1 | 6 | . | 0 | . | 1 | | % | A | c | c | e | e | s |
| *p | | | | | | | | | | | | | * | | | | | | | |
| Pro | Filler | | Source | | | | | | | | | | | | | | | | | |

### Obtain the syslogtag

Behind the IP it is a blank followed by a percent sign. The percent indicates that the syslogtag is following. To move the pointer to the syslogtag position once again a ‘Character Match’ rule is necessary. It has to match the space (actual position of the pointer) and the percent sign. This content is not needed therefore assign it to the ‘Filler’ property.

A colon is immediately behind the syslogtag. So all characters between the percent sign and the colon are needed. The ‘UpTo’ type can do this job. Insert an ‘UpTo’ rule. As value enter ‘:’ (without the quotes) and select the syslogtag property. Note that after parsing the pointer stands on the first character of the ‘UpTo’ value.




- Post-Process Editor*

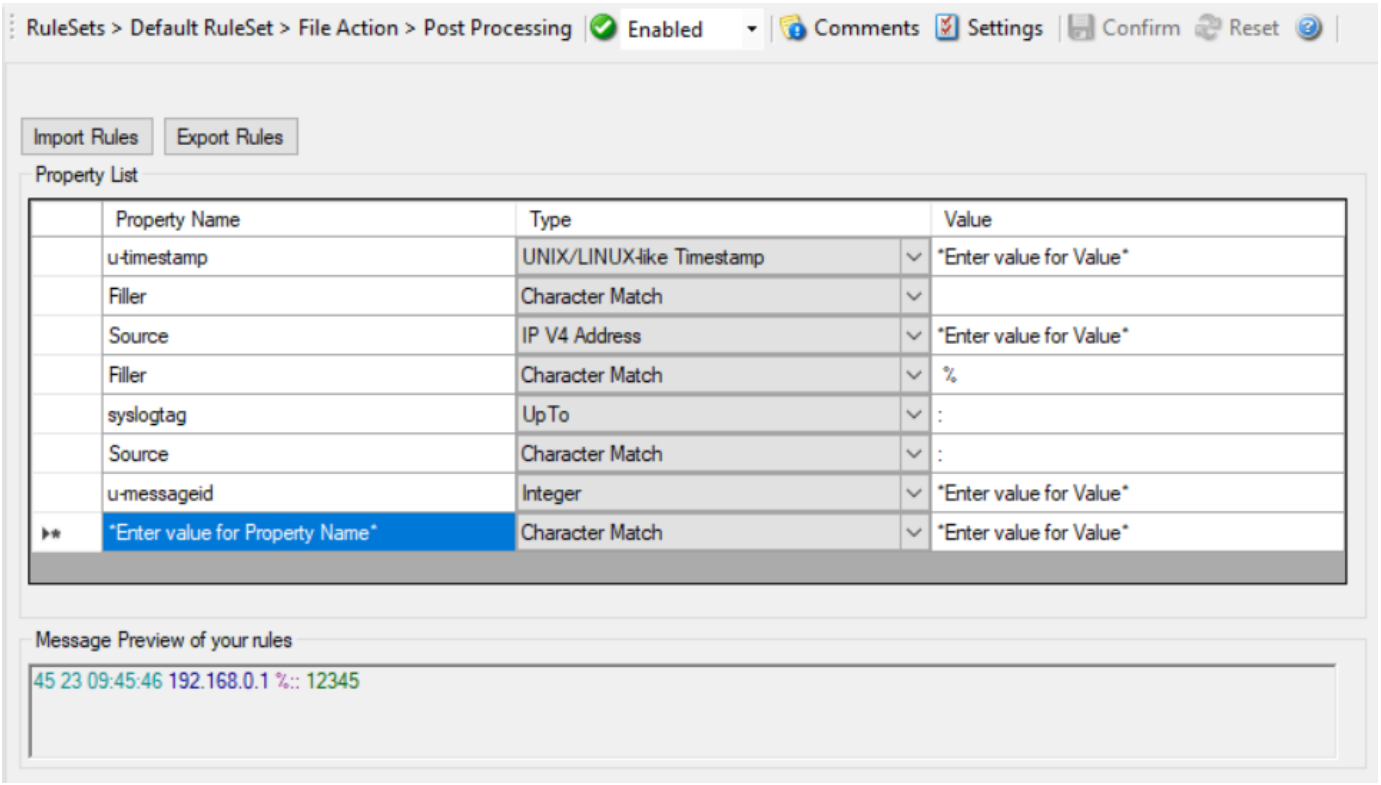
| Pos | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Log | l | | & | A | c | c | e | s | s | - | U | s | e | r | : | | l | 2 | 3 | 4 |
| *p | | | | | | | | | | | | | | | * | | | | | |
| Pro | | Filler | | syslogtag | | | | | | | | | | | | | | | | |

- Important: It points to the colon not to the blank.*

### Take the MessageID

The next interesting item is the MessageID. Move the pointer to start position of the MessageID part. Again, do this by using a ‘Character Match’ rule. Keep in mind that the pointer points to the colon. Behind the colon is a space and then the MessageID starts. Thus, the value of the rule has to be ‘: ‘.

MessageID consist of numbers only. For numeric parsing the ‘Integer’ parse type exist. This type captures all characters until a non-numeric character appears. The pointer is moved behind the number. Note that numeric values with decimal dots cannot be parsed with this type (because they are not integers). This means trying to parse 1.1 results in 1, because the dot is a non-numeric value.



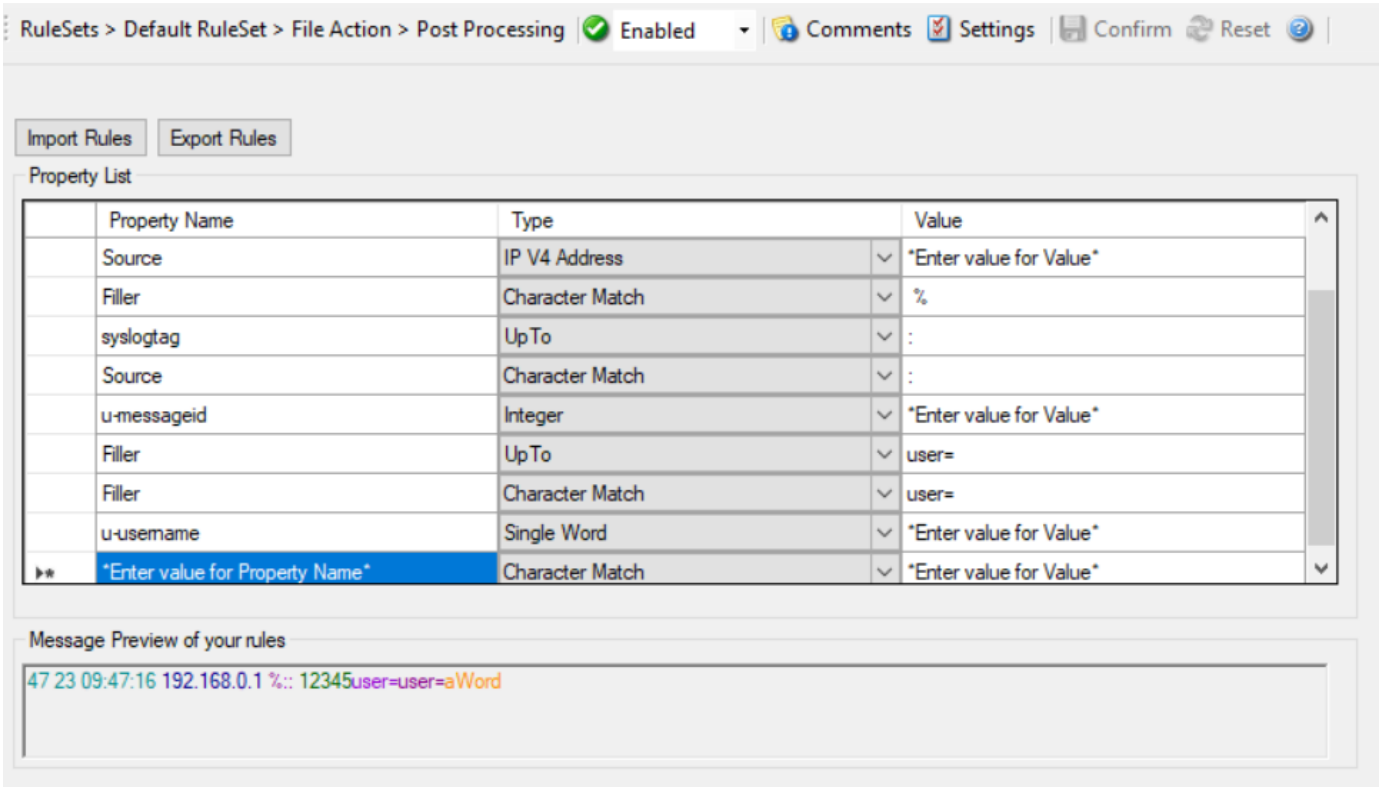

- Post-Process Editor*

| Pos | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Log | r | : | | 1 | 2 | 3 | 4 | 5 | : | | r | u | l | e | = | m | o | n | i | t |
| *p | | | | | | | | | * | | | | | | | | | | | |
| Pro | | | | u-messageid | | | | | | | | | | | | | | | | |

### Find the username and status

Looking at the remainder of the message indicates that the username is not immediately after syslogtag. Thankfully though, the username always starts with ‘user=’. Consequently the ‘UpTo’ type can be used to identify the username. To get the start position of the username we have to use ‘UpTo’ together with ‘Character Match’. Remember that ‘UpTo’ points to the first character of the given value. For this reason the ‘Character Match’ rule is necessary.

After locating the start position of the username ‘Word’ parse type can be used. ‘Word’ parses as long as a space sign is found. Enter ‘u-username’ as property.

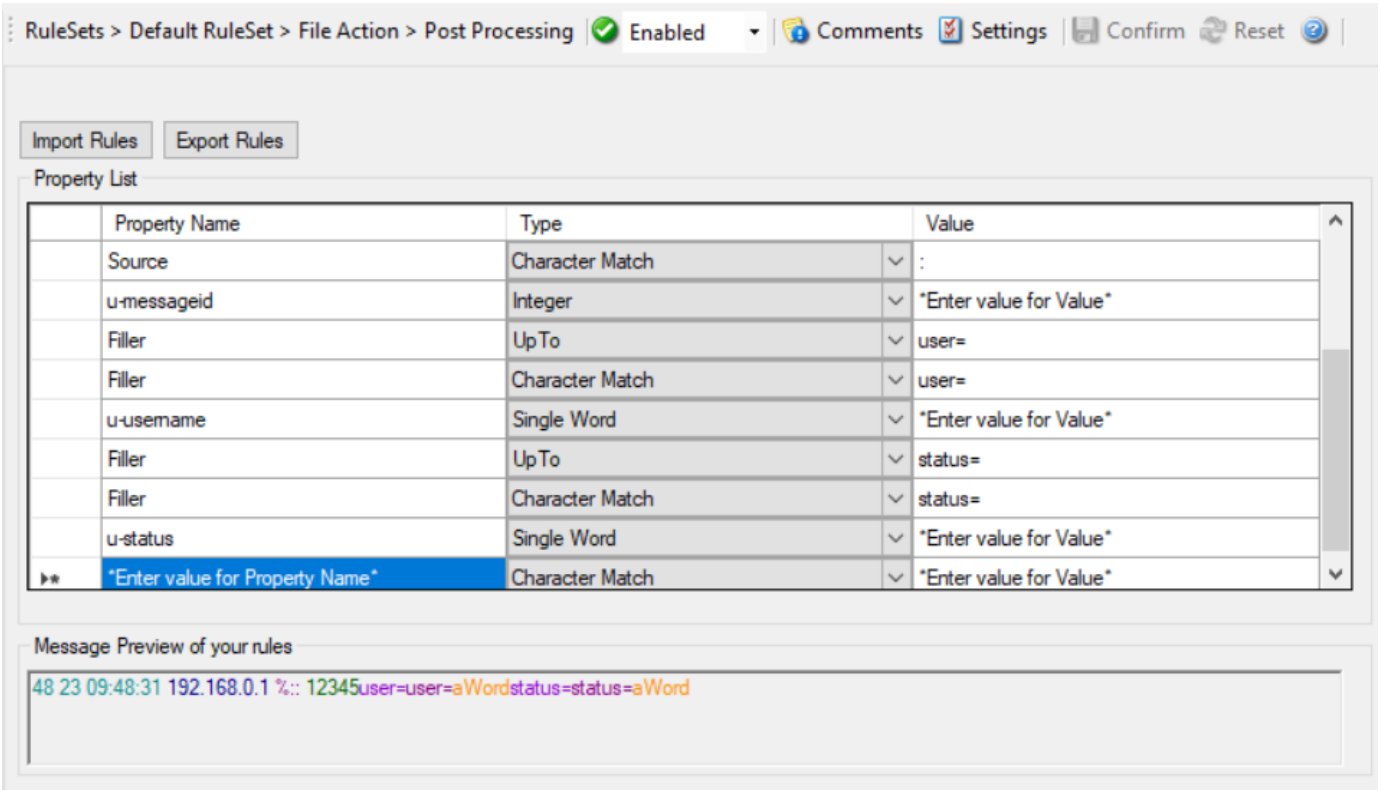



- Post-Process Editor*

| Pos | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Log | i | n | | u | s | e | r | = | B | o | b | | s | t | a | t | u | s | = | d |
| *p | Filler | | | Filler | | | | | u-username | | | * | | | | | | | | |
| Pro | | | | | | | | | | | | | | | | | | | | |

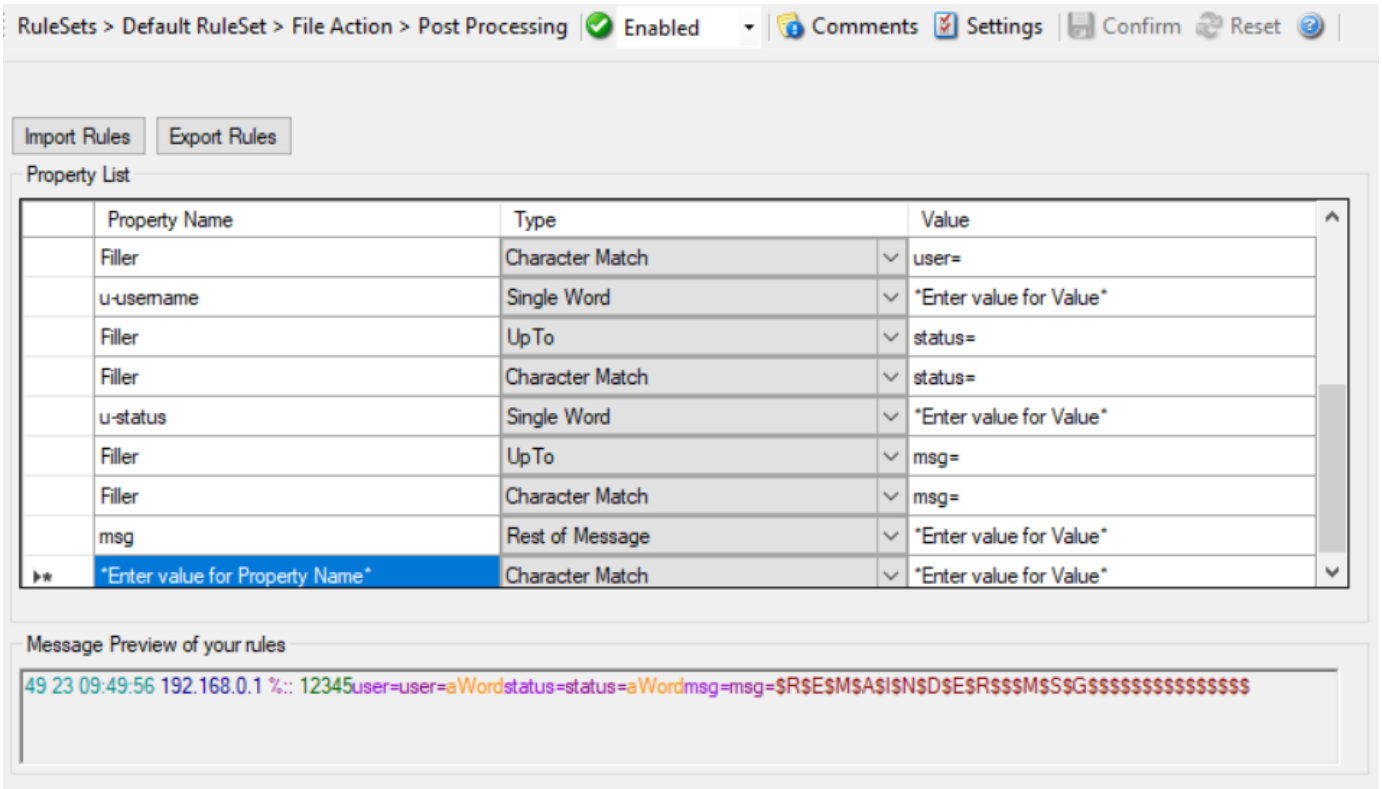
- Notice: After parsing a word the pointer stands on the space behind the parsed word.*

The steps to get the status are very similar to the previous one




### The last rule - Additional Information

One item of interest is left. The last part of the message contains additional information. It starts after ‘msg=’. So the combination of ‘UpTo’ and ‘Character Match’ is used to go to the right position. All characters after ‘msg=’ until the end of the message are interesting. For this purpose the ‘Rest of Message’ parse type is available. It stores all characters from the current position until the end of the message. This also means that this rule can only be used once in a template and is always the last rule.

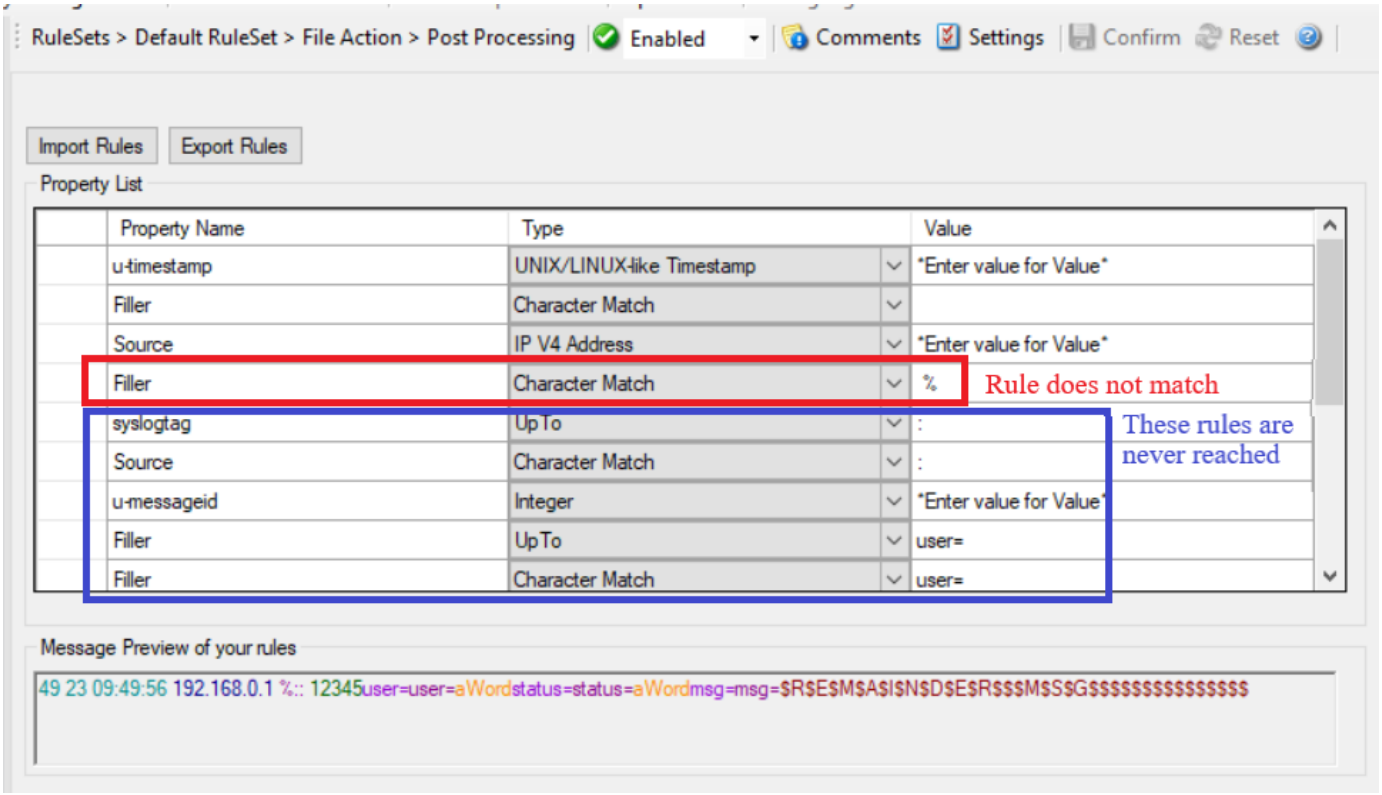



- Complete parse template.*

### What happens if the parser fails?

If a rule does not match processing stops at this point. This means all properties of rules which were processed successfully until the non-matching rule occurs are available.

Let's assume the fourth rule of the following sample does not match.




The first three rules were processed successfully. Therefore u-timestamp and Source are available. But syslogtag and u-messageid are always empty due to the parser never process this rules.

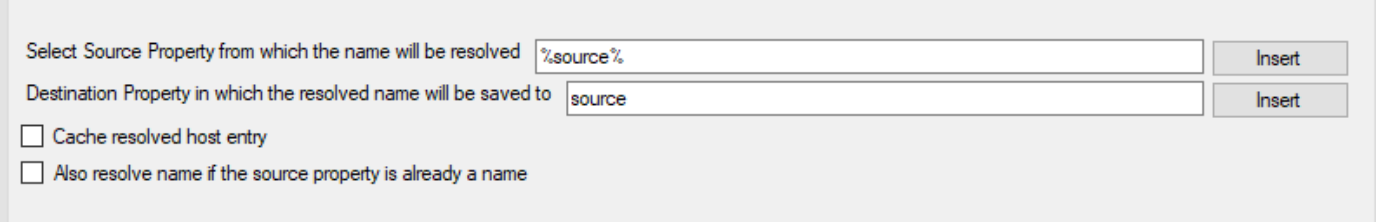
[1] Using the "u-" prefix is recommended to differentiate between MonitorWare-defined properties and user defined one. It is not required, but often of great aid. A common trap is that newer versions of MonitorWare may use property names that a user has also used. MonitorWare will never use any name starting with "u-", so the prefix also guards against such a scenario.

[2] Filler is a predefined property which acts as a bin for unwanted characters. Essentially, the data is simply discarded.

Please Note: There's also a StepByStep Guide available which describes how the PostProcessAction works, you can find it here.

### Resolve Hostname Action

The **Resolve Hostname** action resolves a host name or address from one event property and stores the result in another property. Because it is an action, you can place it in the ruleset where the resolved value is needed.




- Action - Resolve Hostname*

#### Select Source Property from which the name will be resolved

**File Configuration field:**

szSourcePropertyName

**Description:**

Click on the Insert menu link on the right side of the textfield to customize the source property from which the name will be resolved.

#### Destination Property in which the resolved name will be saved to

**File Configuration field:**

szDestinationPropertyName

**Description:**

Same as above, please click on the Insert menu link on the right side of the textfield to customize the destination property in which the resolved name will be saved to.

#### Also resolve name if the source property is already a name

**File Configuration field:**

nResolveIfName

**Description:**

Activates the feature that the name will also be resolved if there is already a source property with that name.

#### Cache resolved host entry

**File Configuration field:**

nCacheNameEntry

**Description:**

If activated this will, as it says, cache the resolved host entry.

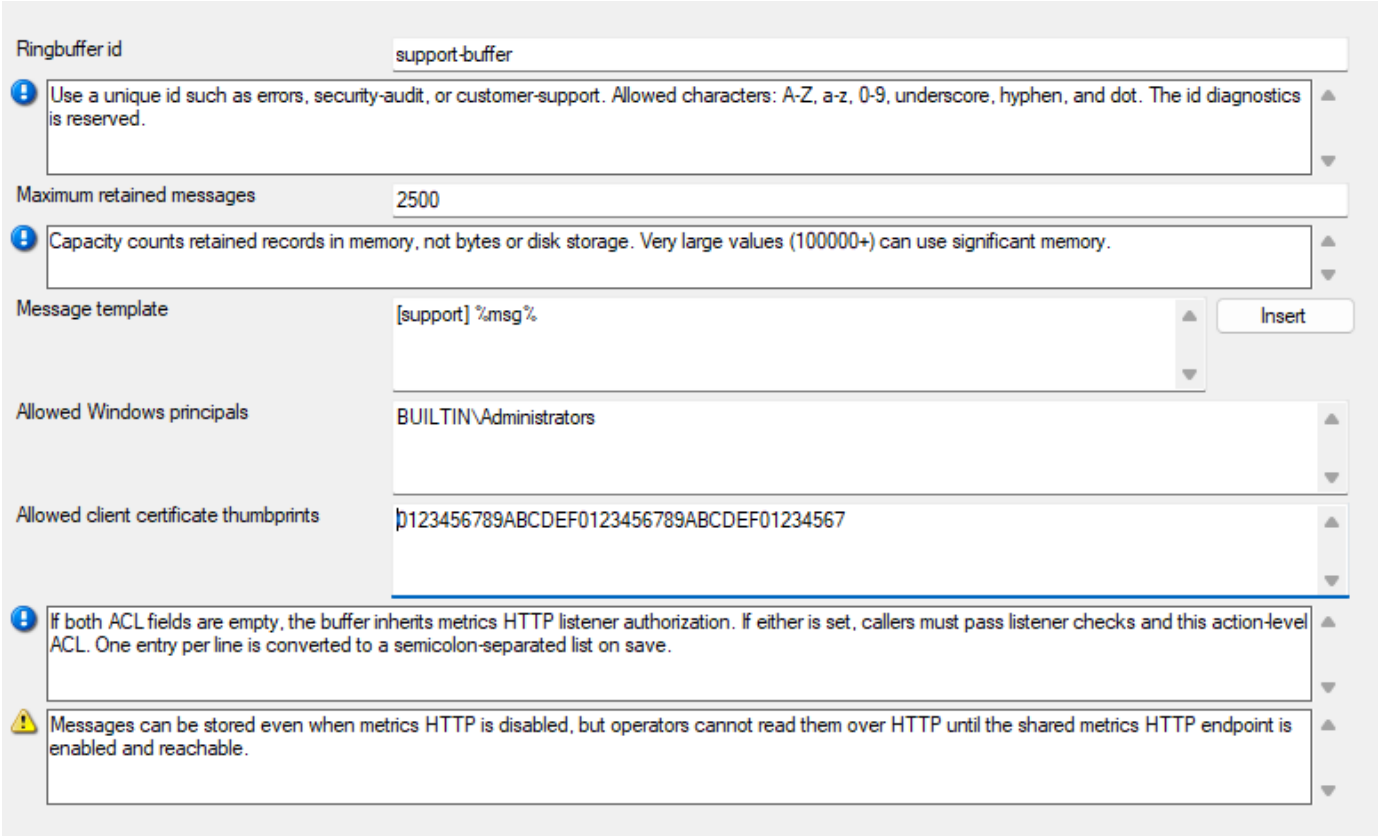
### Message Ringbuffer Action

The **Message Ringbuffer** action keeps recent copies of processed messages in a named, in-memory buffer. Use it for bounded troubleshooting or an approved integration that needs recent messages. It does not replace archival logging, audit retention, or a disk-backed queue.

The action is available starting with **26.08** in WinSyslog, WinSyslog, EventReporter, and rsyslog Windows Agent builds when the installed edition includes it.

Messages are retained only while the service and this action configuration remain active. When the buffer is full, the oldest retained messages are overwritten. Its contents are lost when the service restarts, when a configuration reload recreates the action, or when the action is removed.

Remote reads use the secure metrics HTTP listener. Enable that listener through the file-based `general(name="Metrics")` settings; see **`metrics-http-listener`**. Current configuration clients preserve the Metrics block but do not provide a Metrics editor.




*Message Ringbuffer action properties.*

#### Ringbuffer ID

**File Configuration field:**

szRingBufferId

**Description:**

Enter a unique ID for this buffer. Use one to 64 letters, digits, periods, underscores, or hyphens. The ID identifies the configured buffer, so do not reuse it for another Message Ringbuffer action. The ID `diagnostics` is reserved and cannot be used.

#### Maximum retained messages

**File Configuration field:**

nRingBufferMaxMessages

**Description:**

Set the number of messages to retain, from 2 through 1,000,000. This is a record count, not a storage size. Choose a bounded value after considering the average message size and the available memory. Larger buffers keep more history but use more memory.

### Message template

**File Configuration field:**

szMessageTemplate

**Description:**

Define the text stored for each message. The default is `%msg%`. Use the property insertion control to include the event properties needed by the intended troubleshooting or integration workflow.

### Allowed Windows principals

**File Configuration field:**

szRingBufferAllowedWindowsPrincipals

**Description:**

Optionally list the Windows users, groups, aliases, or SID strings that may read this buffer. Separate entries with commas, semicolons, or line breaks. Listener authorization is inherited only when both action ACL fields are empty. If either this field or **Allowed client certificate thumbprints** is populated, the caller must still pass the metrics HTTP listener checks and must also match at least one populated action ACL list.

### Allowed client certificate thumbprints

**File Configuration field:**

szRingBufferAllowedClientCertificateThumbprints

**Description:**

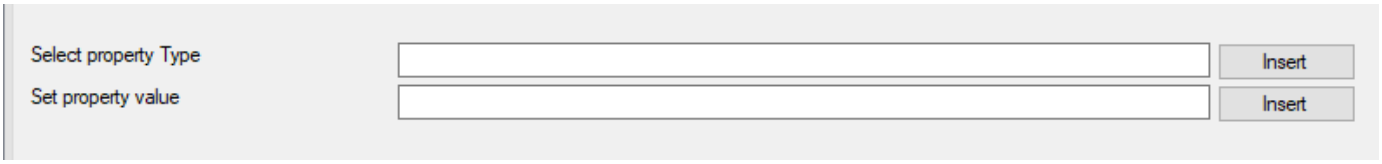
Optionally list SHA-1 client certificate thumbprints that may read this buffer. Do not enter SHA-256 fingerprints; this field matches SHA-1 only. Use this only when the deployment requires client-certificate authentication on the configured secure metrics listener. Listener authorization is inherited only when both action ACL fields are empty. If either this field or **Allowed Windows users, groups, aliases, or SID strings** is populated, the caller must still pass the metrics HTTP listener checks and must also match at least one populated action ACL list.

### Set Property

You can set every property and custom properties using this action.

This dialog controls the set property options. With the “Set Property” action, some properties of the incoming message can be modified. This is especially useful if an administrator would like to e.g. rename two equally named devices.

**Please note: when you change or create a property, the value will be changed as soon as the set property action is carried out. It will not change before that happens and the old value is no longer available thereafter. That means all actions and filter conditions will use the new value after it is set. So, if you would like e.g. rename a system, make sure the set property actions are at the top of the rule base!**




- Action - Set Property*

### Select Property Type

**File Configuration field:**

szPropertyType

**Description:**

Select the property type to be changed. The list box contains all properties that can be changed. By default it is set to nothing.

Please note that the field content can be configured with event properties are described in the property replacer section.

### Set Property Value

**File Configuration field:**

szPropertyValue

**Description:**

The value to be assigned to the property. Any valid property type value can be entered.

Please note that the field content can be configured with event properties are described in the property replacer section.

### Difference from Set Status

*Set Property* changes data on the current message. The updated value is then used by later filters and actions for that same message only.

Use *Set Property* when you need to rewrite, normalize, or enrich message content. Use Set Status instead when you need a global value that persists across multiple messages.

### Set Status

Each information unit have specific properties e.g. EventID, Priority, Facility etc. These properties have some values. Lets suppose that EventID has property value 01. Now, If you want to add “a new property of your own choice” in the existing set of properties then Set Status action allows you to accomplish this!

You can create a new property and assign any valid desired value to it e.g. we create a new property as CustomerID and set its value to 01. After you have created the property through this action, then you can define filters for them. There is an internal status list within the product which you can use for more complex filtering.

Please note: when you change a property, the value will be changed as soon as the set status action is carried out. It will not change before that happens and the old value is no longer available thereafter. That means all actions and filter conditions will use the new value after it is set. So if you would like e.g. rename a system, make sure the set status actions are at the top of the rule base!




- Action - Set Status*

#### Status Variable Name

**File Configuration field:**

szPropertyName

**Description:**

Enter the Property name. That name will from now on be used inside the rule base. More precisely, it will be used in the filter conditions and actions.

Please note that the field content can be configured with event properties are described in the property replacer section.

#### Status Variable Value

**File Configuration field:**

szPropertyValue

**Description:**

The value to be assigned to the property. Any valid property type value can be entered.

Please note that the field content can be configured with event properties are described in the property replacer section.

#### Difference from Set Property

*Set Status* and *Set Property* look similar, but they solve different problems:

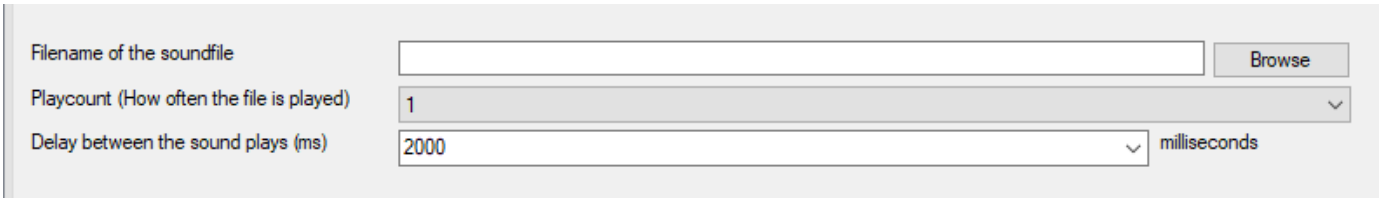
- A **property** belongs to the current message and exists only while that message is processed.
- A **status variable** is global and remains available across multiple messages until another action changes it.

Use *Set Status* when you need shared state across messages, for example a counter, workflow state, or a flag that later filters can evaluate.

## Other actions

### Play Sound

This action allows you to play a sound file. Since Windows VISTA/2008/7, Microsoft has disabled any interaction between a system service and the user desktop. This includes playing sounds as well. So if you want to use the Play Sound Action on any of this Windows Version, you will need to run the service in console mode (From command prompt with the -r option).




- Action - Play Sound*

**Please note:** if your machine has multiple sound cards installed, the **Play Sound** action uses the playback device that Windows exposes as the primary output for the service context.

If you need a different playback device, run the service under a user account whose Windows audio settings select the desired primary output. This is an advanced workaround and is usually of limited practical value on modern server systems.

#### Filename of the Soundfile

**File Configuration field:**

szFilename

**Description:**

Please enter the name of the sound file to play. **This must be a .WAV** file, other formats (like MP3) are not supported. While in theory it is possible that the sound file resides on a different machine, we highly recommend using files on the local machine only. Using remote files is officially not supported (but currently doable if you are prepared for some extra effort in getting this going).

If the file can either not be found or is not in a valid format, a system beep is emitted instead (this should - by API definition - be possible on any system).

#### Playcount

**File Configuration field:**

nCount

**Description:**

This specifies how many times the file is played. It can be re-played up to a hundred times.

#### Delay between Plays

**File Configuration field:**

nDelay

**Description:**

If multiple repeats are specified, this is the amount of time that is to be waited for between each individual play.

### Start Program

With the “Start Program” action, an external program can be run. Any valid Windows executable can be run. This includes actual programs (EXE files), as well as scripts like batch files (.BAT), or VB scripts (.vbs).

Start process can, for example, be combined with the service monitor to restart failed services. Another example application is a script that deletes temporary files if the disk space monitor detects a low space condition.




- Action - Start Program*

### Command to execute

**File Configuration field:**

szCommand

**Description:**

This is the path of actual program file to be executed. This can be the path of any valid executable file. A relative file name can be specified if it can be found via the operating system default search path.

### Use legacy parameter processing

**File Configuration field:**

nUseLegacyProcessing

**Description:**

When enabled, old style parameter processing is used. Otherwise all properties can be used.

### Parameters

**File Configuration field:**

szParameters

**Description:**

These parameters are passed to the program executed. They are passed as command line parameters. There is no specific format – it is up to the script to interpret them.

Parameters can contain replacement characters to customize it with event details. This allows passing event data to the script. The following replacement characters can be used:

- %d Date and time in local time
- %s IP address or name (depending on the “resolve hostnames” setting) of the source system that sent the message.
- %f Numeric facility code of the received message
- %p Numeric priority code of the received message
- %m The message itself
- %% Represents a single % sign.

In the example above, replacement characters are being used. If a message “This is a test” were received from “172.16.0.1”, the script would be started with 3 parameters:

Parameter 1 would be the string “e1” – it is assumed that this has some meaning to the script. Parameter 2 would be the IP address, 172.16.0.1. Parameter 3 would be “This is a test”. Please note that due to the two

quotes (“), the message is interpreted as a single parameters. If they were missing, it would typically be split into several ones, with parameter 3 being “This”, 4 being “is”, and so on. So these quotes are very important!

### Sync Timeout

**File Configuration field:**

nSyncTimeOut

**Description:**

Time Out option is under Sync. Processing. When a program is executed, the service waits for it to finish before it carries on further actions. This is needed in order to ensure that all actions are carried out in the correct sequence.

The external program should only run for a limited amount of time. If it would block for some reason, the agent would be prevented from carrying out any further processing. As such, a timeout value must be specified. If the program still runs after the configured timeout, the rule engine cancels it, flags the action as unsuccessful, and then carries on with processing.

Important: Even though the timeout value can be as high as 30 seconds, we strongly recommend limiting the run time of external program to below 5 seconds. Otherwise, they could affect the overall performance too much. If the average run time is 5 seconds, the default timeout of 10 seconds ensures that the program can finish even when there is high system activity.

For performance reasons, we also strongly recommend to use the “Start Program” action only for rules that apply relatively seldom.

# FAQ

Use this section for direct answers to common WinSyslog questions. The entries below are grouped by user intent so you can scan by task instead of by product component.

## Operation and troubleshooting

### Why were dynamic log files sometimes not rotated in older WinSyslog versions?

Older WinSyslog versions could miss some scheduled rotations when dynamic filenames were combined with file-handle cleanup. That historical behavior was improved in later versions of the shared log rotation subsystem.

For the guarantees of the current subsystem, including graceful shutdown, restart recovery, reload behavior, and dynamic filename support, see **`../../shared/faq/log-rotation-guarantees`**.

#### Background

In older WinSyslog versions, the automatic file handle cleanup logic could close an inactive dynamic log file before the scheduled rotation time arrived. Later versions improved the shared log rotation subsystem to handle this more reliably.

#### The Problem

Users may experience inconsistent log file rotation behavior where:

- Some log files rotate successfully every day as scheduled
- Some log files rotate only partially (not every day)
- Some log files never rotate at all

This typically occurs when log rotation is scheduled at specific times (e.g., at 0:00 every day).

#### Root Cause

The issue is related to WinSyslog’s Automatic File Handle Cleanup feature, which by default:

- Closes inactive log files after 2 hours of no activity

- Frees up memory and system resources
- Prevents resource exhaustion when many log files are being monitored

At the time of scheduled rotation, if a log file has been inactive for more than 2 hours, WinSyslog may have already closed its file handle. When the rotation process runs, it cannot rotate a file that is no longer actively opened by WinSyslog.

**Note:** This behavior is similar to how your computer closes unused programs to maintain system stability.

### Affected Versions

This issue affects WinSyslog versions 17.5 and earlier. The log rotation logic was redesigned starting with WinSyslog version 18.1, which resolves these limitations.

### Solutions

**Recommended Solution: Upgrade WinSyslog**

The most effective solution is to upgrade to WinSyslog version 18.1 or later, where the log rotation logic has been redesigned to handle these scenarios properly.

**Alternative Solution: Adjust File Handle Cleanup Interval**

If upgrading is not immediately possible:

1. Open the WinSyslog configuration
2. Navigate to the service settings
3. Increase the “Automatic File Handle Cleanup” interval from the default 2 hours to 24 hours (or longer, depending on your environment)
4. This adjustment will reduce the likelihood of missing log rotations

**Important:** The longer cleanup interval may increase memory usage, so monitor your system’s resource utilization accordingly. .. _log-rotation-file-handle-cleanup-winsyslog:

## Why were dynamic log files sometimes not rotated in older WinSyslog versions?

Older WinSyslog versions could miss some scheduled rotations when dynamic filenames were combined with file-handle cleanup. That historical behavior was improved in later versions of the shared log rotation subsystem.

For the guarantees of the current subsystem, including graceful shutdown, restart recovery, reload behavior, and dynamic filename support, see **`../../shared/faq/log-rotation-guarantees`**.

If you are still running one of the older affected builds, increasing the file handle cleanup timeout can reduce the chance of a missed scheduled rotation, but the recommended long-term solution is to upgrade to a current release.

## Log Rotation Naming Convention Change in WinSyslog 18.x

This article explains the change in rotated log file naming convention in WinSyslog 18.x and later versions.

### Question

Why are my rotated log files named differently after upgrading to WinSyslog 18.x?

### Answer

WinSyslog 18.x and later versions use a new naming convention for rotated log files. Instead of placing sequence numbers before the file extension (e.g., `syslog1.csv`, `syslog2.csv`), the new format appends sequence numbers after the file extension (e.g., `syslog.csv.1`, `syslog.csv.2`).

**This change is intentional and by design.** The new format follows Unix/POSIX conventions and provides better compatibility with common log management tools and scripts.

### What Changed

**Old Format (WinSyslog versions before 18.x):**

- `syslog.csv` (active log file)

- `syslog1.csv` (first rotated file)
- `syslog2.csv` (second rotated file)
- `syslog3.csv` (third rotated file)

**New Format (WinSyslog 18.x and later):**

- `syslog.csv` (active log file)
- `syslog.csv.1` (first rotated file)
- `syslog.csv.2` (second rotated file)
- `syslog.csv.3` (third rotated file)

### Root Cause

This change was intentionally implemented as part of improvements to the log rotation subsystem. The new format provides several benefits:

1. **Better compatibility:** Follows Unix/POSIX conventions used by standard log rotation utilities
2. **Improved reliability:** Enhanced thread safety in the log rotation mechanism
3. **Tool compatibility:** Works better with common log management tools and scripts
4. **Industry standard:** Aligns with widely-adopted log rotation naming practices

**Important:** The filename format change cannot be reverted through configuration settings.

### Impact on Existing Workflows

The naming convention change may affect:

- **Scripts that parse log filenames:** Scripts expecting the old format may fail to find rotated files
- **Monitoring tools:** Tools that reference specific filename patterns may need updates
- **Log management workflows:** Automated processes that depend on the old naming convention may break
- **Backup scripts:** File backup routines that filter by filename pattern may need adjustment

### Solution

#### Updating Scripts and Tools

If you have scripts, monitoring tools, or applications that parse or reference rotated log files, update them to work with the new format:

1. **Modify file parsing logic:**

   Update patterns to handle `filename.ext.N` format instead of `filenameN.ext`

   Example (PowerShell):

   ```
   # Old pattern
   Get-ChildItem "syslog[0-9].csv"

   # New pattern
   Get-ChildItem "syslog.csv.[0-9]"
   ```

2. **Update hardcoded filename references:**

   Replace any hardcoded file paths in scripts to use the new naming convention
3. **Test compatibility:**

   Verify script functionality with the new naming convention in a test environment before deploying to production

#### Log Management Tool Compatibility

The new format is compatible with most modern log rotation utilities:

- **Standard logrotate:** Fully compatible with the new format
- **Third-party tools:** Most log management tools support POSIX-style naming

- **Custom solutions:** May require updates to filename matching patterns

**Recommendation:** Verify that your log management tools support the POSIX-style naming convention. Most modern tools do, but older or custom solutions may need configuration updates.

### Migration Best Practices

When upgrading to WinSyslog 18.x or later:

1. **Test in development:**
   - Deploy the new version in a development environment first
   - Run your log processing workflows and scripts
   - Verify all tools work correctly with the new naming format
   - Document any required changes
2. **Update automation:**
   - Modify scripts before deploying the new WinSyslog version
   - Update monitoring tool configurations
   - Test all changes in the development environment
3. **Plan for transition:**
   - Consider running both old and new versions during a transition period
   - Update scripts to handle both naming conventions if needed during migration
   - Document the change in deployment procedures
4. **Verify backup processes:**
   - Ensure backup scripts include files with the new naming pattern
   - Test backup restoration to verify rotated files are included
   - Update retention policies if they depend on filename patterns
5. **Update documentation:**
   - Document the filename format change in maintenance procedures
   - Update runbooks and operational guides
   - Communicate changes to all stakeholders

### Common Questions

**Can I configure WinSyslog to use the old naming format?**

No. The new naming format is built into the log rotation subsystem and cannot be changed through configuration. This ensures consistent behavior and maintains the reliability and thread safety improvements.

**Will my existing rotated log files be renamed automatically?**

No. Existing rotated files retain their original names. The new naming convention applies only to files rotated after upgrading to WinSyslog 18.x or later.

**What happens to old rotated files?**

Old rotated files (using the previous naming convention) remain unchanged. They coexist with newly rotated files that use the new convention. You may want to rename old files manually if consistency is important for your workflows, or simply let them age out according to your retention policy.

**Are there any performance implications?**

No. The naming convention change does not affect performance. In fact, the underlying improvements to the log rotation subsystem provide better reliability and thread safety.

### Additional Information

For more information about log rotation configuration, see the log file action documentation in the WinSyslog manual.

If you need assistance updating scripts or tools to work with the new naming convention, contact Adiscon support at https://ticket.adiscon.com/

## Why does log rotation fail when using ZIP compression in WinSyslog?

This article explains why log rotation preparation can time out before the service detaches a file for background compression or move processing.

### Problem

Log rotation preparation can fail when the “Maximum wait time for log rotation” setting is too short for the File Action to acquire the rotation mutex while the service is busy.

### Symptoms

- Log files are compressed into ZIP format but remain in the live logging directory
- Rotation is skipped or deferred because the rotation preparation mutex could not be acquired in time
- Incomplete log rotation leaves compressed files in active directories
- Current day logs may be archived prematurely when using time-based rotation triggers

### Root Cause

The “Maximum wait time for log rotation” setting in the WinSyslog Configuration Client controls how long rotation preparation waits to acquire the File Action rotation mutex. Compression and move operations run afterward as detached background work. When this wait is too short, the file may not be detached for rotation under load.

### Solution

#### Option 1: Increase Maximum Wait Time for Log Rotation

1. Open the WinSyslog Configuration Client
2. Navigate to the rotation settings section
3. Locate the “Maximum wait time for log rotation” setting
4. Change the value from 15000 (15 seconds) to 60000 (60 seconds) or 120000 (120 seconds)
5. Save the configuration and restart the WinSyslog service
6. Test log rotation during the next scheduled rotation period

#### Option 2: Switch to Size-Based Rotation

1. Change rotation trigger from time-based to size-based
2. Configure appropriate file size thresholds for rotation
3. Adjust rotation timing to avoid current-day log pre-archiving
4. Test rotation behavior with smaller log files first

### Best Practices

- Set the rotation wait time to at least 60 seconds when the service is busy or storage is slow
- Consider increasing to 120 seconds for large log files or slow storage systems
- Use size-based rotation instead of time-based to prevent current-day log pre-archiving

### Related Settings

- **Maximum wait time for log rotation**: Controls how long rotation preparation waits for the File Action rotation mutex (in milliseconds)
- **Rotation trigger**: Determines when rotation begins (time-based vs size-based)
- **Compression method**: Affects processing time (ZIP compression takes longer than other methods)

### Verification

- Monitor log directories after rotation triggers to ensure files are properly moved

- Check that compressed files are not remaining in live logging directories
- Verify rotation completes within expected timeframes
- Confirm no rotation failures in WinSyslog logs during test periods

## Are WinSyslog products affected by recent OpenSSL CVEs?

### Question

Are WinSyslog products affected by recent OpenSSL CVEs? Which OpenSSL version do the products use, and are the vulnerable components used?

### Problem

Customers may see OpenSSL security advisories (e.g., multiple CVEs from OpenSSL 3.6, 3.5, 3.4, 3.3, 3.0, or 1.1.1/1.0.2 branches) and need to know:

- Whether WinSyslog is affected by specific CVEs
- Which OpenSSL version is shipped with WinSyslog
- Whether the vulnerable code paths or components are used

### Symptoms

- Security or compliance teams request a formal assessment of OpenSSL CVEs for WinSyslog
- Scans or reports may flag WinSyslog due to bundled OpenSSL
- No observable runtime failure; this is a security/compliance assessment topic

### Solution

WinSyslog **26.07** bundles an updated OpenSSL library as part of the platform refresh. OpenSSL advisories list affected version ranges per CVE. Many CVEs affect only certain release branches (for example 3.6, 3.5, 3.4, 3.3, 3.0, 1.1.1, 1.0.2) and do **not** include every minor line shipped with Adiscon products.

If WinSyslog ships OpenSSL from a branch that is not in the affected set for a given CVE, the product is not vulnerable to that CVE regardless of whether the vulnerable API exists in the code base.

**Information:**

- OpenSSL versions are embedded into the product statically without dependencies on system-installed versions
- The product uses its own bundled OpenSSL library, independent of any OpenSSL installation on the system
- This means system OpenSSL updates do not affect the product, and conversely, the product’s OpenSSL does not affect system security

**Important Notes:**

- OpenSSL version information for your specific build can be obtained from Adiscon Support
- Adiscon monitors security advisories and provides updates as necessary
- For the most current information, consult the WinSyslog release notes or contact Support

### Notes

## Troubleshooting the Start Program action in WinSyslog

This article explains common issues with the Start Program action in WinSyslog and provides solutions to resolve them.

### Background

The Start Program action allows WinSyslog to execute external programs, batch files, or scripts when specific conditions are met. However, there are several common issues that can prevent this action from working correctly.

## Common Issues and Solutions

**Issue 1: Program not found or path problems**

**Symptoms:** - The Start Program action appears to run but nothing happens - No error messages in the Windows Event Log - The external program works when run manually from command line

**Root Cause:** WinSyslog may not be able to locate the executable file due to path issues or missing dependencies.

**Solutions:**

1. **Use absolute paths for all executables** - Instead of: `curl google.com > temp.txt` - Use: `C:\curl\curl-win\bin\curl.exe google.com > C:\temp\temp.txt`
2. **Verify executable location** - Check if the program exists in the specified path - Ensure all required DLL files are present - Test the command manually from Windows Command Prompt
3. **Check Windows PATH environment variable** - WinSyslog may not have access to the same PATH as your user session - Use full paths instead of relying on PATH resolution

**Issue 2: Permission problems**

**Symptoms:** - No error messages in Event Log - Program works when run manually but not through WinSyslog

**Root Cause:** WinSyslog runs as a Windows service with different permissions than your user account.

**Solutions:**

1. **Store files in accessible locations** - Avoid system folders like `C:\Windows\System32` - Use generic folders like `C:\temp` or `C:\scripts` - Ensure WinSyslog service has read/execute permissions
2. **Check file permissions** - Right-click on the executable file - Go to Properties > Security - Ensure “SYSTEM” and “SERVICE” accounts have execute permissions

**Issue 3: Working directory problems**

**Symptoms:** - Program runs but cannot find input/output files - Relative paths in scripts don’t work

**Root Cause:** The working directory when WinSyslog executes the program may be different from expected.

**Solutions:**

1. **Use absolute paths for all file references** - Instead of: `> temp.txt` - Use: `> C:\temp\temp.txt`
2. **Set working directory in batch files** - Add `cd /d C:\your\working\directory` at the beginning of batch files

## Troubleshooting Steps

1. **Check Windows Event Log** - Open Event Viewer (type “Event Viewer” in Windows search) - Navigate to Windows Logs > Application - Look for WinSyslog-related error events
2. **Test with simple commands first** - Start with a basic batch file that creates a text file - Example: `echo Test > C:\temp\test.txt`
3. **Verify the command works manually** - Open Command Prompt as Administrator - Run the exact same command that WinSyslog should execute - Ensure it works from the command line first
4. **Check WinSyslog service account** - Verify which account WinSyslog is running under - Ensure that account has necessary permissions

## Example Working Configuration

Here’s an example of a properly configured Start Program action:

**Command to execute:** `C:\curl\curl-win\bin\curl.exe`

**Parameters:** `google.com > C:\temp\response.txt`

**Key points:** - Full path to curl.exe - Absolute path for output file - No reliance on PATH environment variable - Output directory exists and is writable

## Additional Tips

- **Timeout settings:** Keep external programs under 5 seconds runtime for best performance
- **Error handling:** Consider adding error checking to your batch files

- **Logging:** Add logging to your scripts to help troubleshoot issues
- **Testing:** Always test Start Program actions in a development environment first

If you continue to experience issues after following these steps, please contact Adiscon support with: - WinSyslog version - Windows version - Exact command being executed - Any error messages from Event Log - Results of manual command testing

## High-load configuration reload issues in WinSyslog

This article explains why WinSyslog configuration reloads can stall under heavy load and how to stabilize reload behavior.

### Problem

During periods of high message volume, WinSyslog detects configuration changes but may not complete the reload process. The service can appear to hang during reloads and stop requests can time out.

### Symptoms

- Configuration changes are detected, but Event ID 126 (“Configuration reload successfully done”) is not logged
- Service stop operations timeout with “Could not stop the service within 20 seconds” errors
- Reloads complete successfully during low-load periods but fail during high-load periods
- Noticeable time gaps between detection of a configuration change and reload completion

### Root Cause

A WinSyslog configuration reload requires the service to pause message processing, drain in-flight work, reload configuration data, reinitialize rules and actions, and then resume processing. Under high message volume, draining in-flight work takes much longer when worker threads are insufficient, which stretches the reload window and can prevent completion.

### Solution

#### Option 1: Increase worker threads

1. Open the WinSyslog Configuration Client.
2. Navigate to **General Options > Queue Manager**.
3. Set **Number of worker threads** to at least half the CPU core count.
    - Example: For an 8-core system, set at least 4 worker threads.
    - Example: For a 16-core system, set at least 8 worker threads.
4. Save the configuration and allow WinSyslog to reload.

#### Option 2: Upgrade to the latest build

Upgrade to the latest WinSyslog build to pick up fixes that address configuration reload and log rotation behavior under load.

#### Option 3: Reduce the main Queue Limit when Action Queue is enabled

If you use Action Queue on database actions, review **General Options > General > Queue Limit** and keep it around 100,000 to 200,000 when configured higher. This reduces queue management overhead while Action Queue continues to buffer output.

#### Option 4: Disable automatic reload in production

If configuration changes are infrequent, clear **Automatically reload service on configuration changes** in **General Options > General** and perform manual service restarts during maintenance windows.

### Option 5: Split configuration into multiple services and rulesets

A single syslog service with a single ruleset forces every message to be evaluated against all rules. Split the configuration so each input uses a smaller ruleset.

Implementation steps:

1. Create additional syslog services with distinct ports and permitted senders.
2. Create matching rulesets and move rules to the appropriate ruleset.
3. Update device destinations to use the new ports.
4. Update firewall rules for the new ports.
5. Test each service and ruleset combination.
6. Migrate one network segment at a time.
7. Keep the original configuration as a backup until migration is complete.

Performance impact:

- Before: one service, one ruleset, all rules evaluated per message.
- After: multiple services and rulesets, fewer rules per ruleset, fewer evaluations per message.
- Expected: 4-8x throughput improvement under high load.

### Best Practices

- Set worker threads to at least half the CPU core count.
- Use Action Queue for database actions instead of expanding the main queue.
- Plan reloads during low-traffic periods or scheduled maintenance windows.
- Monitor Event ID 126 to confirm reload completion.

### Verification

1. Check the Windows Application Event Log for Event ID 126 after configuration changes.
2. Confirm WinSyslog service stop operations complete within the timeout period.
3. Monitor worker thread utilization and queue depth during peak load.
4. Verify reload completion times are consistent under normal conditions.

## Queue Buildup During SQL Server Table Cleanup Operations in WinSyslog

This article explains queue buildup issues in WinSyslog when performing regular cleanup operations on Microsoft SQL Server SystemEvents tables.

### Question

Why does WinSyslog's message queue build up when deleting old rows from the SQL Server SystemEvents table, even though the table is not explicitly locked?

### Answer

When using Microsoft SQL Server as storage via OLEDB or ODBC Actions in WinSyslog, performing regular cleanup operations (deleting old rows) on the SystemEvents table may cause WinSyslog's message queue to build up even though the table is not explicitly locked. This can occur even with optimized batch delete processes that use primary key-based deletes.

**Note:** This issue applies specifically to WinSyslog using Microsoft SQL Server as storage through OLEDB or ODBC Actions. The queue buildup occurs when cleanup operations (DELETE statements) run concurrently with WinSyslog's INSERT operations to the same SQL Server database.

### Symptoms

- Queue saturation incidents that correlate with cleanup schedule times
- Queue buildup during delete operations, even with batch delete processes
- Brief blocking or slowdowns during cleanup operations

- High memory consumption when queue holds large numbers of messages
- Processing rate barely keeping up with ingestion rate during cleanup

### Root Cause

Even with optimized delete processes, several subtle mechanisms can cause contention between INSERT and DELETE operations:

1. **Page-Level Locks**: SQL Server may use page-level locks during deletes. If WinSyslog's INSERT operations target the same pages being deleted, brief blocking can occur. With high-frequency inserts (200-400 messages per second or higher), even brief page-level contention can cause queue buildup.
2. **Index Maintenance Overhead**: The primary key index must be maintained during each delete batch. Even with efficient primary key-based deletes, index pages need to be updated, which can cause brief contention that slows INSERT operations.
3. **Transaction Log Activity**: Regular delete operations generate significant transaction log activity. If the transaction log is on the same disk as data files, I/O contention can occur during delete operations, temporarily slowing all database operations including inserts.
4. **Ghost Record Cleanup**: SQL Server marks deleted rows as "ghost records" initially, then cleans them up asynchronously. If ghost record cleanup coincides with high INSERT activity, page-level contention can occur.

These issues are more subtle than traditional locking problems and may not show up as explicit table locks, but can still cause queue buildup during delete operations.

### Solution

#### Option 1: Enhance Delete Process with ROWLOCK Hint

The ROWLOCK hint forces SQL Server to keep locks at the row level instead of escalating to page-level locks, which means INSERT operations can proceed on other rows within the same pages even while deletes are running. Adding OPTION (MAXDOP 1) prevents parallel execution that could escalate locks.

**Implementation:**

```
DELETE TOP (5000) FROM SystemEvents WITH (ROWLOCK)
WHERE [Date] < DATEADD(day, -1, GETDATE())
OPTION (MAXDOP 1);
```

**Benefits:**

- Simple change that can reduce blocking
- No SQL Server edition changes required
- Works with Standard Edition
- Reduces lock escalation to page level

**Considerations:**

- Provides partial improvement, not complete elimination of contention
- Still requires index maintenance during deletes

#### Option 2: Separate Transaction Log Disk

Place the transaction log on a separate physical disk, separate from data files. This is a SQL Server best practice and eliminates I/O contention between log writes and data file operations.

**Benefits:**

- Eliminates I/O contention between log and data operations
- Works with any SQL Server edition
- Best practice for SQL Server configuration

**Considerations:**

- Requires separate physical disk or storage volume
- Addresses I/O contention only, not locking issues

### Option 3: Table Partitioning (Enterprise Edition)

Partition the SystemEvents table by date (e.g., daily partitions). Inserts always go to today's partition while deletes target old partitions that no longer receive inserts, completely eliminating contention. When cleaning up, use partition switching to instantly move an entire partition to a staging table and drop it - this is a metadata-only operation that takes milliseconds instead of minutes.

**Benefits:**

- Completely eliminates contention between inserts and deletes
- Partition switching is a metadata-only operation (milliseconds)
- Inserts and deletes operate on different partitions

**Considerations:**

- Requires SQL Server Enterprise Edition
- Requires initial setup and ongoing partition management
- More complex implementation

### Option 4: Delayed Durability (If Acceptable Risk)

This can improve insert performance by 20-40% by deferring transaction log writes. However, there's a risk of losing the last few seconds of inserts if the server crashes before the log is flushed.

**Implementation:**

```
ALTER DATABASE [DatabaseName] SET DELAYED_DURABILITY = ALLOWED;
```

**Benefits:**

- Significant performance improvement (20-40%)
- Works with Standard Edition
- Simple configuration change

**Considerations:**

- Risk of data loss if server crashes before log flush
- Addresses I/O performance only, not locking issues
- May not be acceptable for all environments

## Best Practices

1. **Monitor Queue Saturation Correlation**: Check whether queue saturation incidents correlate with cleanup schedule times. If deletes run on a regular schedule and queue buildup occurs at predictable intervals, this strongly suggests the cleanup process is a contributing factor.
2. **Monitor SQL Server Blocking**: Use the following query to check for blocking during delete operations:

```
SELECT
r.session_id,
r.blocking_session_id,
r.wait_type,
r.wait_time,
SUBSTRING(t.text, 1, 200) AS QueryText
FROM sys.dm_exec_requests r
CROSS APPLY sys.dm_exec_sql_text(r.sql_handle) t
WHERE r.blocking_session_id > 0;
```

3. **Optimize Batch Delete Process**: Use primary key-based deletes with appropriate batch sizes (e.g., 5000 rows) to balance performance and lock duration. Using the primary key ensures efficient index seeks.
4. **Isolate Parsing Workload**: If using separate processes for parsing, use (NOLOCK) hints to completely isolate the parsing workload from WinSyslog's insert operations.
5. **Verify Transaction Log Location**: Ensure transaction log is on separate physical disk from data files to eliminate I/O contention.

6. **Consider Action Queue Feature**: Use Action Queue feature at Database Action level (OLEDB or ODBC Actions) instead of increasing main queue limit excessively. This can help manage queue buildup during temporary database slowdowns.

### Related Settings

- **Main Queue Limit**: The maximum number of messages that can be queued in WinSyslog. Large limits (e.g., 2-4 million) can cause increased CPU overhead and memory consumption when queue is full.
- **Worker Threads**: Number of worker threads for parallel processing in WinSyslog. Increasing worker threads (e.g., from 2 to 4) can improve parallel processing during normal operations.
- **Action Queue**: Feature at Database Action level (OLEDB or ODBC Actions) in WinSyslog that provides additional buffering during temporary database slowdowns. Recommended over excessive main queue limits.
- **Database Connection Settings**: Connection timeout and retry settings for SQL Server connections via OLEDB or ODBC Actions in WinSyslog.

### Verification

To verify if cleanup process is contributing to queue issues in WinSyslog:

1. **Check Timing Correlation**: Monitor whether queue saturation incidents occur at predictable intervals matching cleanup schedule.
2. **Monitor Blocking**: Run the blocking query during cleanup operations to identify any blocking sessions.
3. **Review SQL Server Wait Statistics**: Check for PAGEIOLATCH, LCK_M_* wait types during cleanup operations.
4. **Monitor Transaction Log Activity**: Check transaction log file growth and I/O during cleanup operations.
5. **Review Queue Metrics**: Monitor queue depth over time in WinSyslog and correlate with cleanup schedule to identify patterns.

If queue buildup incidents correlate with cleanup schedule times, the recommendations described above can help address the contention issues.

## Default Timevalues Setting in WinSyslog Explained

### Overview

This FAQ explains the Default Timevalues setting in WinSyslog and how it affects time handling throughout the system.

### What is the Default Timevalues Setting?

The General Options in WinSyslog contain a setting for “Default Timevalues”. This setting controls how timestamps are handled throughout the system.

### Default Timevalues Options

You can set Default Timevalues to:

- **UTC (Universal Coordinated Time):** The default setting. All internal time calculations use UTC.
- **Localtime:** Uses the local time zone for certain output operations.

### Why UTC is Used Internally

**Important:** Even when you select “Localtime”, WinSyslog still calculates internally with UTC time. This is necessary to:

- Maintain time accuracy when messages are sent via Syslog or SETP protocols
- Prevent unexpected time differences across time zones
- Ensure consistent timestamp handling in database operations
- Support reliable message forwarding between systems

### Where Localtime Setting Has Effect

When Default Timevalues is set to “Localtime”, it affects the following operations:

| Action | Effect |
|---|---|
| Send Email Action | Date in the email header uses localtime |
| Start Program Action | Time parameters in command line use localtime |
| Write File Action | Time properties in file names use localtime |
| Filter Engine | Filtering by weekday or time fields uses localtime |

### Getting Localtime Output

For actions where you need localtime output in custom formats, you can use property options.

#### Property Option: localtime

Converts the output of any timestamp into localtime:

```
%timegenerated:::localtime%
```

This will display the time in the local time zone.

#### Property Option: uxLocalTimeStamp

Same output as uxTimeStamp, but uses localtime instead of UTC:

```
%timereported:::uxLocalTimeStamp%
```

This generates a UNIX-style timestamp based on local time.

### Examples

**Using UTC timestamps (default):**

```
%timegenerated%
%timereported:::uxTimeStamp%
```

**Using localtime timestamps:**

```
%timegenerated:::localtime%
%timereported:::uxLocalTimeStamp%
```

### Best Practices

- **Keep UTC as default:** For most scenarios, UTC is recommended for consistency
- **Use localtime option:** Apply the localtime property option only where needed
- **Database storage:** Store timestamps in UTC for accurate querying across time zones
- **Display purposes:** Convert to localtime when displaying to users
- **Log rotation:** Use localtime for file naming if you want daily rotations based on local business hours

### Additional Notes

- The Default Timevalues setting affects defaults only – individual actions can override this
- UTC-based calculations ensure messages maintain accurate timestamps when forwarded
- Consider your environment’s time zone distribution when choosing the default
- Most enterprise deployments prefer UTC for global consistency

### Related Topics

For more information about WinSyslog configuration, see the General Options section and the Actions documentation in your WinSyslog manual.

## How to Export WinSyslog Settings for a Support Call

### Overview

When contacting Adiscon support, you may be asked to provide your WinSyslog configuration. This FAQ explains how to export your WinSyslog settings to a registry file and send them to the support team.

> **Note**
>
> Do **not** use the binary registry file export option. Always use the text-based registry file export described below.

### Step 1: Export Settings

Open the WinSyslog Configuration Client. Go to the **Computer** menu and click **Export Settings to Registry File**.

### Step 2: Save the Registry File

After selecting the export option, a dialog appears where you can choose the file name and location. Enter a descriptive name (for example, `winsyslog-config.reg`) and save the file to a convenient location.

### Step 3: Compress the File

Right-click on the saved registry file and compress (zip) it using your preferred archiving tool. This reduces the file size for email transmission.

### Step 4: Send the File

Send the zipped registry file to the Adiscon support team via the Support Portal.

### Security Considerations

The exported registry file contains your WinSyslog configuration, which may include connection strings, server addresses, or other environment-specific details. We recommend reviewing the contents of the registry file with a text editor before sending it, so you can verify that no sensitive information is included unintentionally.

### See Also

- Installation - WinSyslog installation documentation

## Recommended Service Stop Order for WinSyslog Maintenance

### Question

What is the recommended order for stopping the WinSyslog service during system maintenance or reboots?

### Answer

When performing system maintenance, updates, or planned reboots on a system running WinSyslog, follow a specific shutdown sequence to prevent data loss and ensure a clean shutdown. The WinSyslog service should be stopped **after** any web server and **before** the database server.

### Recommended Stop Order

1. **Stop IIS/Web Server** (if using a web-based log viewer)
2. **Stop WinSyslog Service**
3. **Stop Database Server** (SQL Server, MySQL, etc.)

### Rationale

This sequence ensures:

- **Web connections are closed first:** Prevents new user sessions from accessing the database while WinSyslog is still writing.
- **WinSyslog stops gracefully:** Allows WinSyslog to complete any in-progress writes and flush its queues before the database becomes unavailable.
- **Database closes last:** Ensures all pending transactions from WinSyslog are committed before the database shuts down.

### Stop Commands

You can stop the WinSyslog service using its internal service name `AdisconWINSyslog` in PowerShell or Command Prompt:

**Command Prompt:**

```
net stop w3svc
net stop "AdisconWINSyslog"
net stop MSSQLSERVER
```

**PowerShell:**

```
Stop-Service -Name "w3svc"
Stop-Service -Name "AdisconWINSyslog"
Stop-Service -Name "MSSQLSERVER"
```

### Startup Order

When starting services after maintenance, reverse the order:

1. **Start Database Server**
2. **Start WinSyslog Service**
3. **Start IIS/Web Server**

**Command Prompt:**

```
net start MSSQLSERVER
net start "AdisconWINSyslog"
net start w3svc
```

**PowerShell:**

```
Start-Service -Name "MSSQLSERVER"
Start-Service -Name "AdisconWINSyslog"
Start-Service -Name "w3svc"
```

### Service Name Reference

When managing the WinSyslog service from the command line, use the internal service name:

- **Internal Service Name:** `AdisconWINSyslog`
- **Display Name:** WinSyslog Service

The internal service name remains consistent across installations and should be used in automation scripts for reliability.

### Verifying Service Status

```
Get-Service -Name "AdisconWINSyslog"
```

```
sc query "AdisconWINSyslog"
```

### Best Practices

- **Plan maintenance windows:** Schedule downtime during low-traffic periods to minimize log message loss.
- **Backup database:** Perform a database backup before shutting down services.
- **Verify connections:** After restart, verify that the WinSyslog service started correctly and is writing to the database.
- **Check logs:** Review the Windows Event Viewer for any WinSyslog service errors after restart.
- **Use internal service names:** Always use the internal service name `AdisconWINSyslog` in scripts for reliability.

### Troubleshooting

If the WinSyslog service does not stop gracefully:

- Check for stuck processes in Task Manager.
- Review the Windows Event Viewer for service errors.
- Verify that database connections are properly closed.
- Check service dependencies: `sc qc "AdisconWINSyslog"`
- As a last resort, use force stop: `net stop "AdisconWINSyslog" /y`

## Running WinSyslog on a Windows Cluster Server

### Question

Can WinSyslog run on a Windows Cluster Server, and are there any particular issues to be aware of?

### Answer

Yes. WinSyslog runs on a Windows Cluster node without problems. However, WinSyslog does not include built-in cluster failover support. If a node fails, you must start the WinSyslog service on another node manually or through a cluster script. The steps below explain how to prepare the cluster for this scenario.

### Step 1: Set the Service Startup Type

On every cluster node **except** the primary node, set the WinSyslog service startup type to **Manual**:

1. Open the Windows Service Manager (Start > Control Panel > Administrative Tools > Services).
2. Locate the service named **AdisconWINSyslog**.
3. Right-click the service and select **Properties**.
4. Set **Startup type** to **Manual**.
5. Click **Apply**, then **OK**.

On the primary node, leave the startup type set to **Automatic** so that WinSyslog starts automatically after a reboot.

### Step 2: Mirror the Configuration Between Nodes

WinSyslog stores its configuration in the Windows registry. To replicate a working configuration from one node to another, export it as a registry file and import it on each secondary node:

1. Open the WinSyslog Configuration Client on the primary node.
2. Go to the **Computer** menu.

3. Select **Export Settings to Registry File**.
    - Choose the standard registry format (do **not** select a binary format).
    - Select the correct architecture (Win32 or x64) for your system.
4. Save the `.reg` file to a network share or removable media.
5. On each secondary node, double-click the `.reg` file to import the configuration.

After importing, the secondary node has the same configuration as the primary node. When a failover is needed, start the WinSyslog service on the secondary node using the Services Manager or from the command line:

```
net start "AdisconWINSyslog"
```

### Best Practices

- **Keep configurations in sync.** After every configuration change on the primary node, re-export and re-import the registry file on all secondary nodes.
- **Test failover regularly.** Verify that the WinSyslog service starts correctly and processes messages on each secondary node.
- **Use automation.** Consider a cluster resource script or a scheduled task that starts the WinSyslog service on a secondary node when the primary node becomes unavailable.
- **Verify firewall rules.** Ensure that syslog ports are open on all cluster nodes so that traffic can reach the secondary node after failover.

## Why Does My WinSyslog Database Setup Fail?

### Question

Why does my first WinSyslog database logging setup fail to write rows into SQL Server or another ODBC database?

### Answer

The most common causes are:

- the DSN was created as a user DSN instead of a **System DSN**
- the ODBC driver does not match the WinSyslog service architecture
- the DSN works for the interactive user but not for WinSyslog running under its default `Local System` service account
- the SQL account can connect but cannot insert into the target table
- the table name in WinSyslog does not match the real table name
- the field list still reflects the default schema instead of the custom table
- the rule or service that should receive the test message is not active

### Details

For a first setup, WinSyslog database logging usually fails for one of three reasons:

1. The database connection itself is wrong.
2. The action points at the wrong table or column mapping.
3. The test message never reaches the ruleset that contains the database action.

Use the following checks in order.

### Quick checks

1. Test the ODBC DSN outside WinSyslog first.
    - Open **Data Sources (ODBC)**.
    - Confirm that the DSN is a **System DSN**.
    - Confirm that the driver connects to the intended SQL Server instance and database.

- If the DSN uses Windows authentication, remember that WinSyslog normally runs under the default Windows `Local System` service account unless you changed it.
- A DSN test that succeeds for the interactive admin user does not prove that the WinSyslog service can connect with the same DSN and database access.

2. Test the WinSyslog action connection.

   - Open the **Write to Database** action.
   - Click **Verify Database**.
   - If the verify step fails, fix the DSN or credentials before testing anything else.

3. Confirm the target table name.

   - If you use a custom table, the table name in WinSyslog must match the actual SQL table exactly.
   - If you use the default Adiscon schema, keep the built-in table names.

4. Confirm the field list.

   - A custom table must use a field list that matches the table columns.
   - Remove default-schema rows that do not belong to the custom table.

5. Confirm that the message reaches the ruleset.

   - Make sure the input service is enabled.
   - Make sure the service is bound to the ruleset that contains the database action.
   - Make sure the message you test actually matches the rule conditions.

### Common failure patterns

- **DSN works in Windows but not in WinSyslog**

  This usually means the service cannot see the same DSN definition or the DSN was created for the wrong account scope. Recreate it as a **System DSN**.
- **Verify Database succeeds, but no rows appear**

  This usually means the table name, field list, or ruleset binding is wrong. Recheck the action settings and confirm that the incoming message actually triggers the rule.
- **The DSN test works for the admin user, but WinSyslog still cannot write**

  This usually means the DSN or SQL permissions were only validated for the interactive user account. WinSyslog normally runs under the default Windows `Local System` service account unless you changed it, so a remote SQL Server may see a different account than your admin session.

  Recheck the authentication mode and either grant SQL access to the WinSyslog service account context, change the WinSyslog service to run under an account that already has SQL access, or use SQL authentication instead.
- **Rows appear, but values are missing or truncated**

  This usually means the destination column is too short or the wrong WinSyslog property is mapped. Widen the SQL column or use a shorter property replacer expression.
- **SQL Server rejects the insert**

  This usually means the database account lacks insert permission or the table schema does not match the action's field list. Check both permissions and the column definitions.
- **The setup works until the database is unavailable**

  This usually means you should enable the database action's queue settings so WinSyslog can buffer writes while the database is offline.

### Action path

1. Verify the DSN in Windows ODBC administration.
2. Use **Verify Database** in the WinSyslog action.
3. Confirm the table name and field list.
4. Confirm that the message reaches the correct ruleset.
5. Send the test message again and query the target table in SQL Server.

### Related information

- Quick Start: Write Syslog Messages to Microsoft SQL Server
- Tutorial: Integrate WinSyslog with a Custom Database Schema
- ODBC Database Options
- Which Database Format Should I Use with WinSyslog?
- Queue Buildup During SQL Server Table Cleanup Operations in WinSyslog

## Why Does a WinSyslog Message Show Two Timestamps?

### Question

Why do some WinSyslog log entries show two timestamps?

### Answer

Because the entry can contain both the sender-provided event time and the time when WinSyslog received the message. These serve different purposes and both can be useful.

### Details

A typical received syslog message may contain:

- the timestamp created by the sending device or application
- the timestamp added by WinSyslog when the message arrives

The sender timestamp tells you when the original event happened according to that sender. The WinSyslog timestamp tells you when the message reached the WinSyslog server.

These values are often close, but they can differ when:

- the sending device has an incorrect clock
- the sender buffers messages and sends them later
- network or forwarding delays exist
- an intermediate collector relays the message

#### Interpretation guidance

If you need to understand event chronology on the sender, the sender timestamp is usually the more relevant value. If you need to understand transport delay, collection timing, or queuing effects, the WinSyslog receive timestamp is the more useful one.

If the sender does not include a timestamp at all, WinSyslog's receive time may be the only reliable time reference available.

### Related information

- Default Timevalues Setting in WinSyslog Explained
- Syslog Message Properties

## What Does WinSyslog Event ID 1011 Mean?

### Question

What does Event ID 1011 mean when it appears in the Windows Application Event Log for WinSyslog?

### Answer

Event ID 1011 indicates that WinSyslog could no longer find the last Windows Event Log record it had processed before. This usually means the monitored Windows Event Log was cleared or truncated.

### Details

This matters only when WinSyslog monitors Windows Event Log sources.

To continue processing, WinSyslog resets its internal read position and starts again from the beginning of the available log data. That allows monitoring to continue, but it can also lead to duplicate messages because WinSyslog must resume from a safe point.

### Typical causes

- the monitored Windows Event Log was cleared manually
- the log was truncated because of retention settings or size limits
- the underlying record that WinSyslog expected no longer exists

### Operational effect

The condition is usually informational rather than catastrophic. It tells you that continuity was interrupted and that re-reading may occur.

If you see this repeatedly, review the retention policy of the monitored event log and the polling behavior of the event-monitoring service.

### Related information

Review the WinSyslog service configuration that reads Windows Event Log data and the retention policy of the monitored Windows log sources.

## Why did my WinSyslog update report success when the files were not upgraded correctly?

### Question

Why can an in-place WinSyslog update report success while the installed service, client, or selected features still contain files from the older release?

### Answer

This can happen when Windows Installer retains an incomplete feature state while processing a cross-generation update. The setup wizard can finish and report success even though a later maintenance operation still sees features as advertised, unavailable, or otherwise not fully installed. The product configuration is normally not deleted; the problem is that the expected program files were not refreshed.

### Details

This behavior has been confirmed for direct WinSyslog updates from the 16.x generation to the 18.x generation. Other direct source versions can show the same symptoms, including paths within the adjacent release generations. The source version alone does not prove that an update path is safe or affected.

The underlying mechanism is consistent with Windows Installer feature-state handling. A newer setup package can be accepted as an update without changing the installed product identity, but Windows Installer still evaluates the feature and component state recorded for the previous installation. A change between product generations can leave a feature only partially selected or advertised. `ADDLOCAL=ALL` explicitly requests all features locally, while `REINSTALL=ALL` only refreshes features that Windows Installer considers installed. This is why a two-stage repair is useful for this symptom.

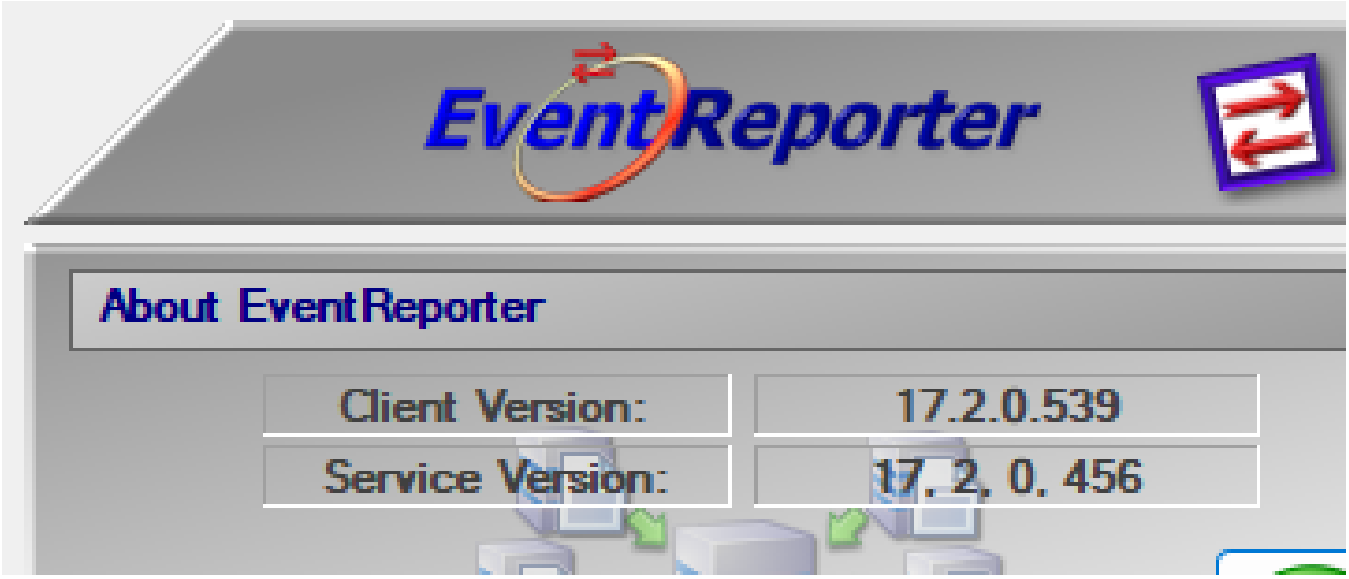
Look for this combination:

- the setup wizard displays a successful completion page
- the WinSyslog About dialog or file properties still show an older client or service version
- Modify or Custom Setup shows an unexpected state such as `0 of 1 subfeatures selected` or `0 KB`
- a normal repair does not replace the older files
- the installer requests a reboot because a service or file is still in use

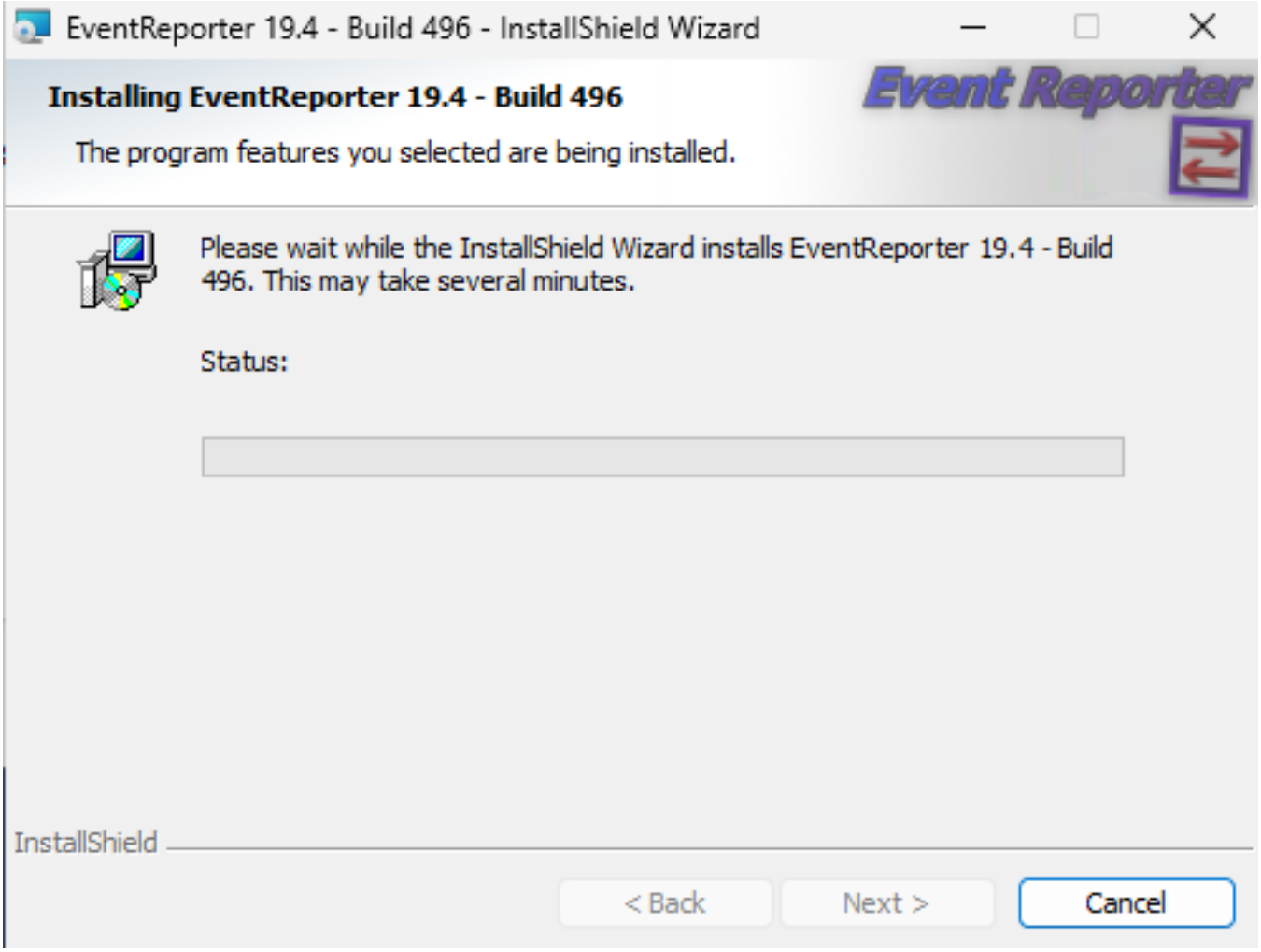
**orphan:**

### Representative setup evidence

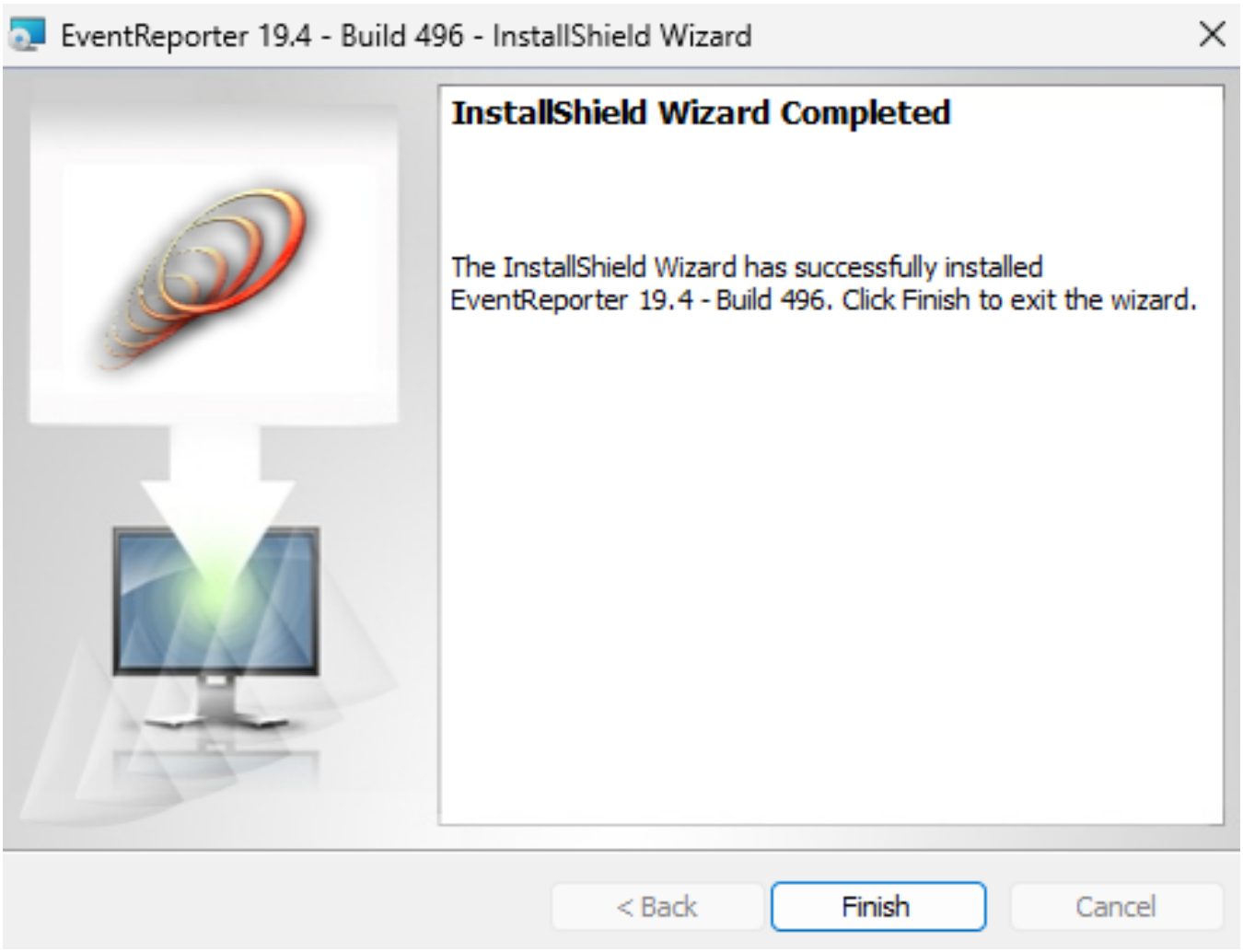
The screenshots below come from one affected setup run and are representative of the visual clues described in this article. Product names, component labels, and version numbers can differ in other product installations.



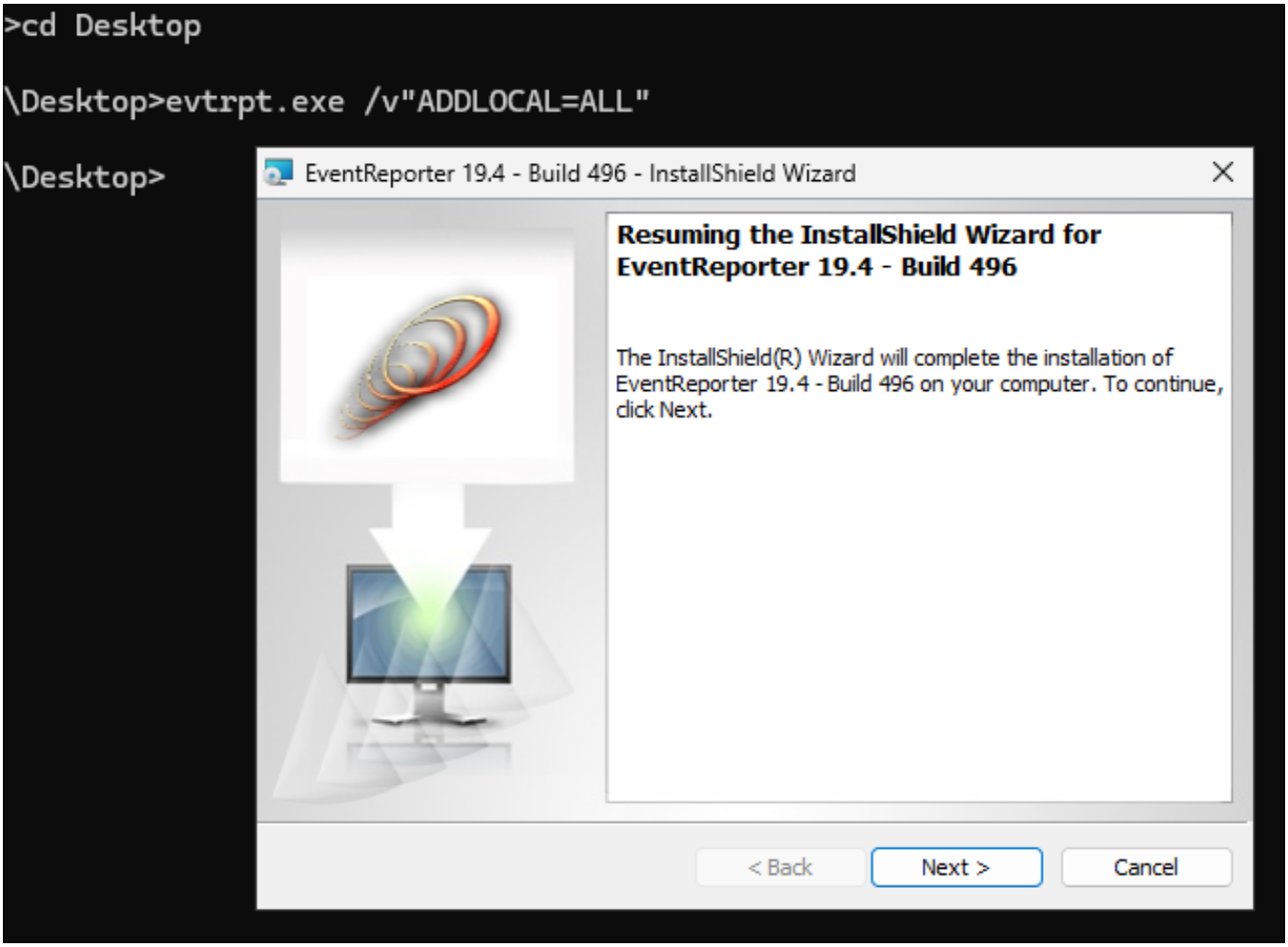

*The installed product reports the older client and service versions before the update.*



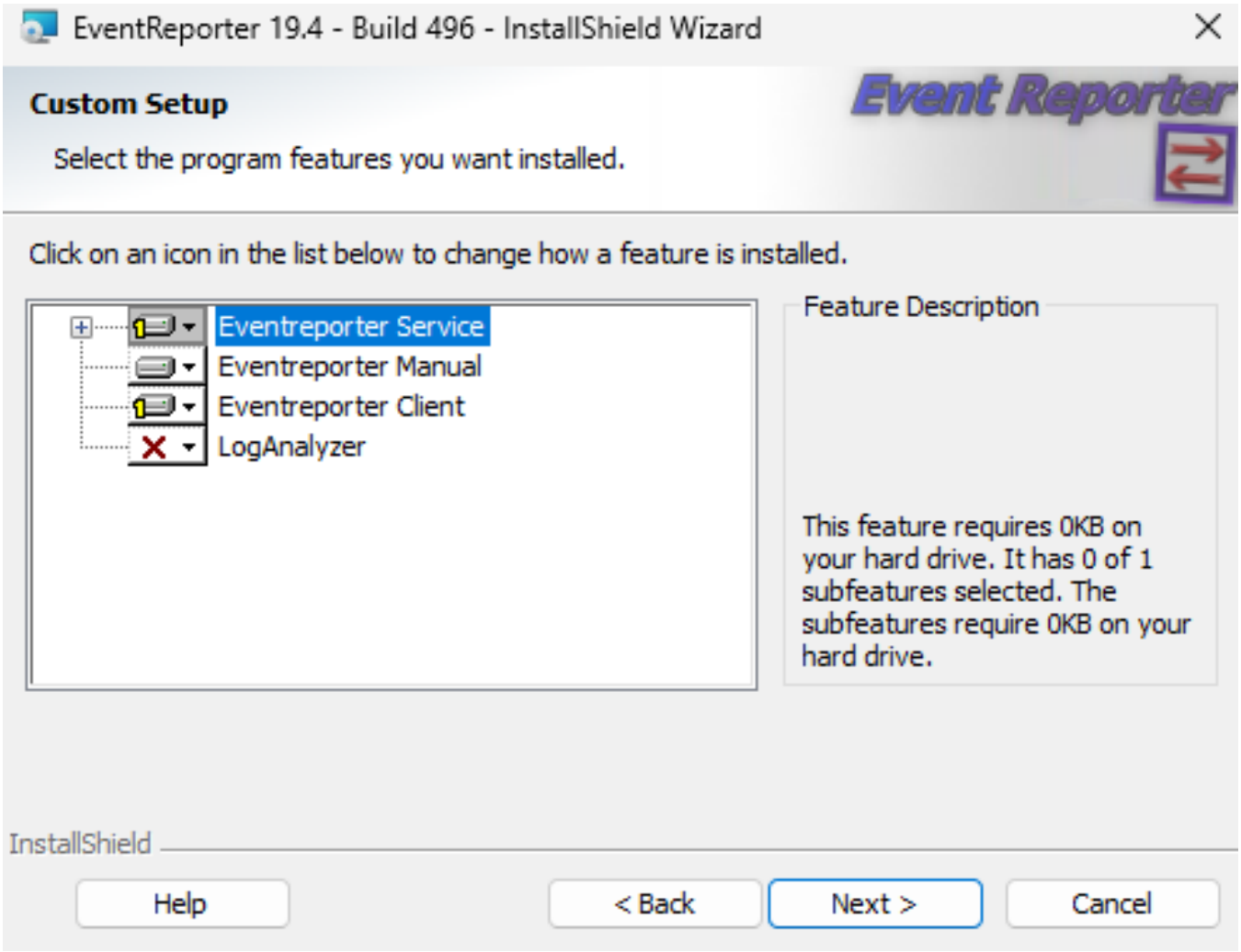

*The setup wizard can display a normal installation-progress page.*



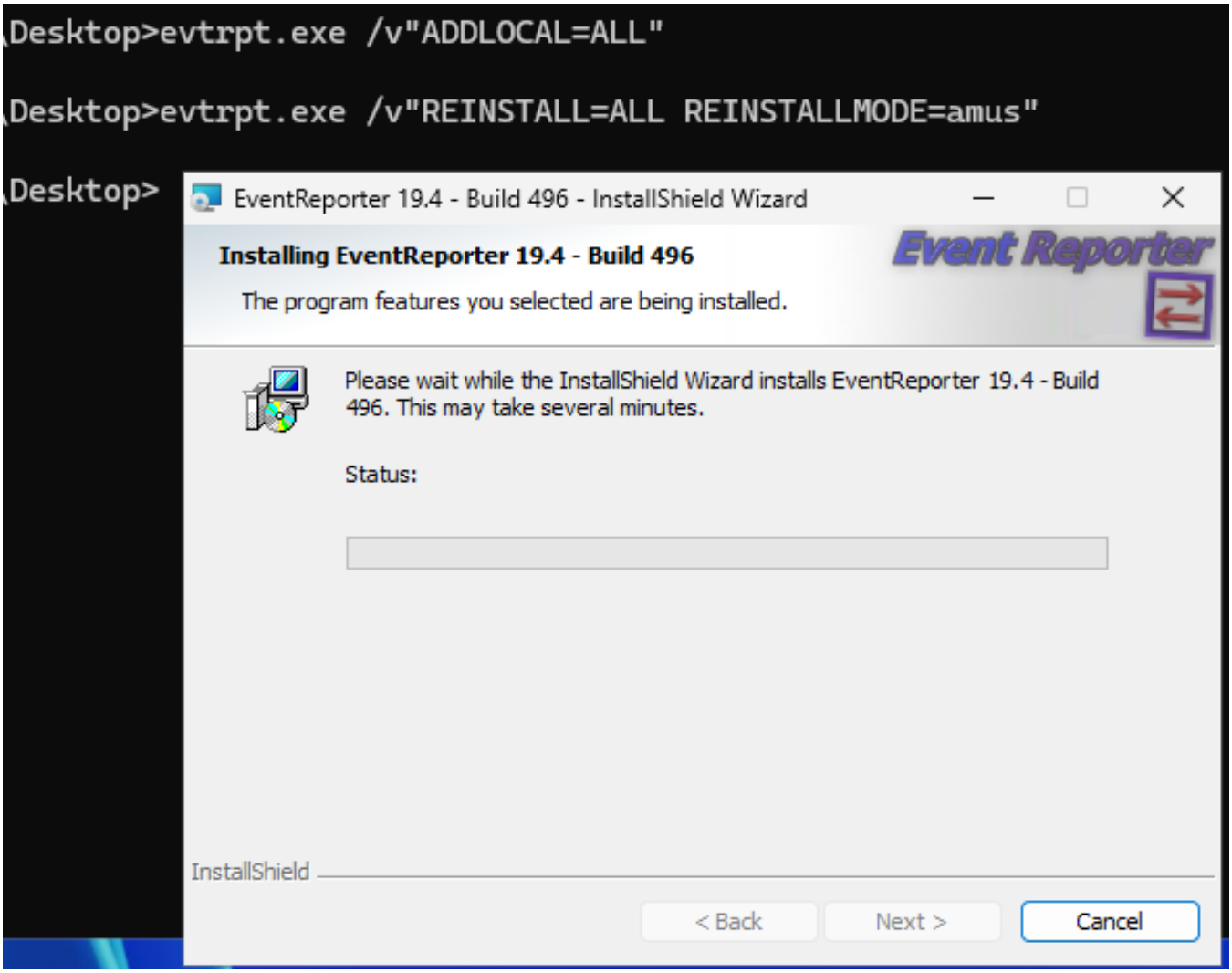

*The wizard can report that the installation completed successfully.*



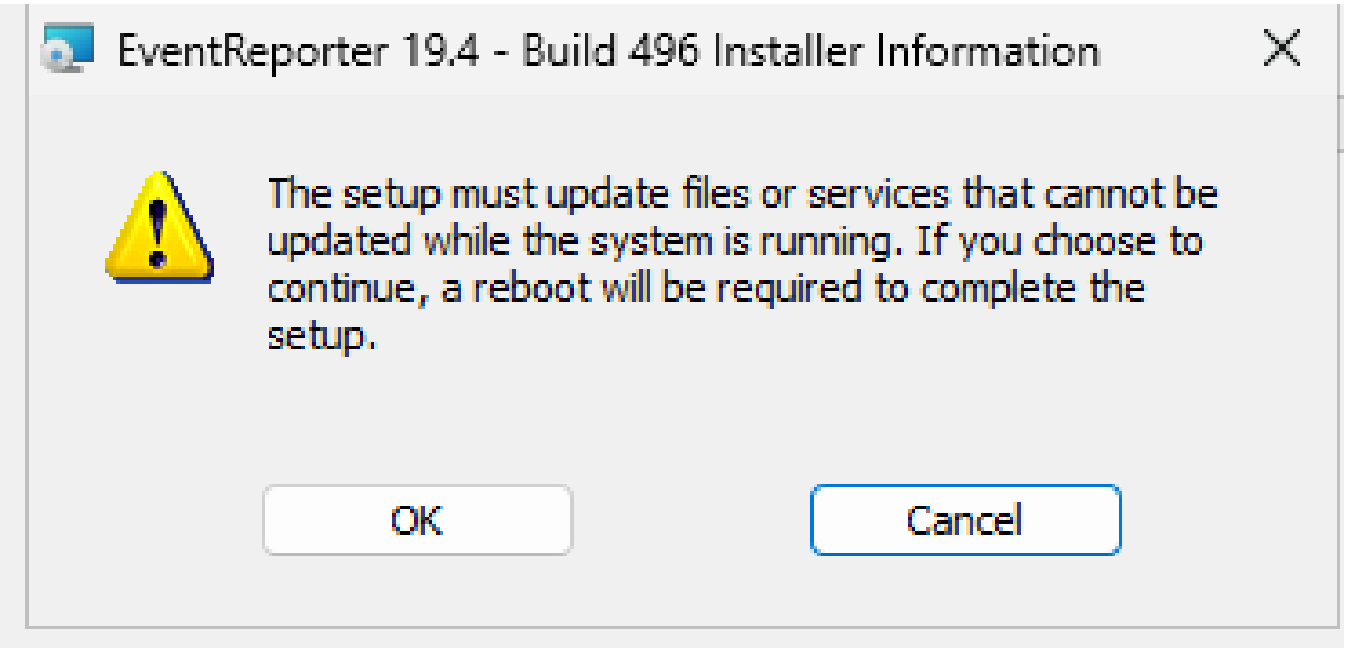

*A later maintenance action can resume the InstallShield wizard.*




*Custom Setup can show an unexpected feature state, such as zero selected subfeatures or zero required disk space.*



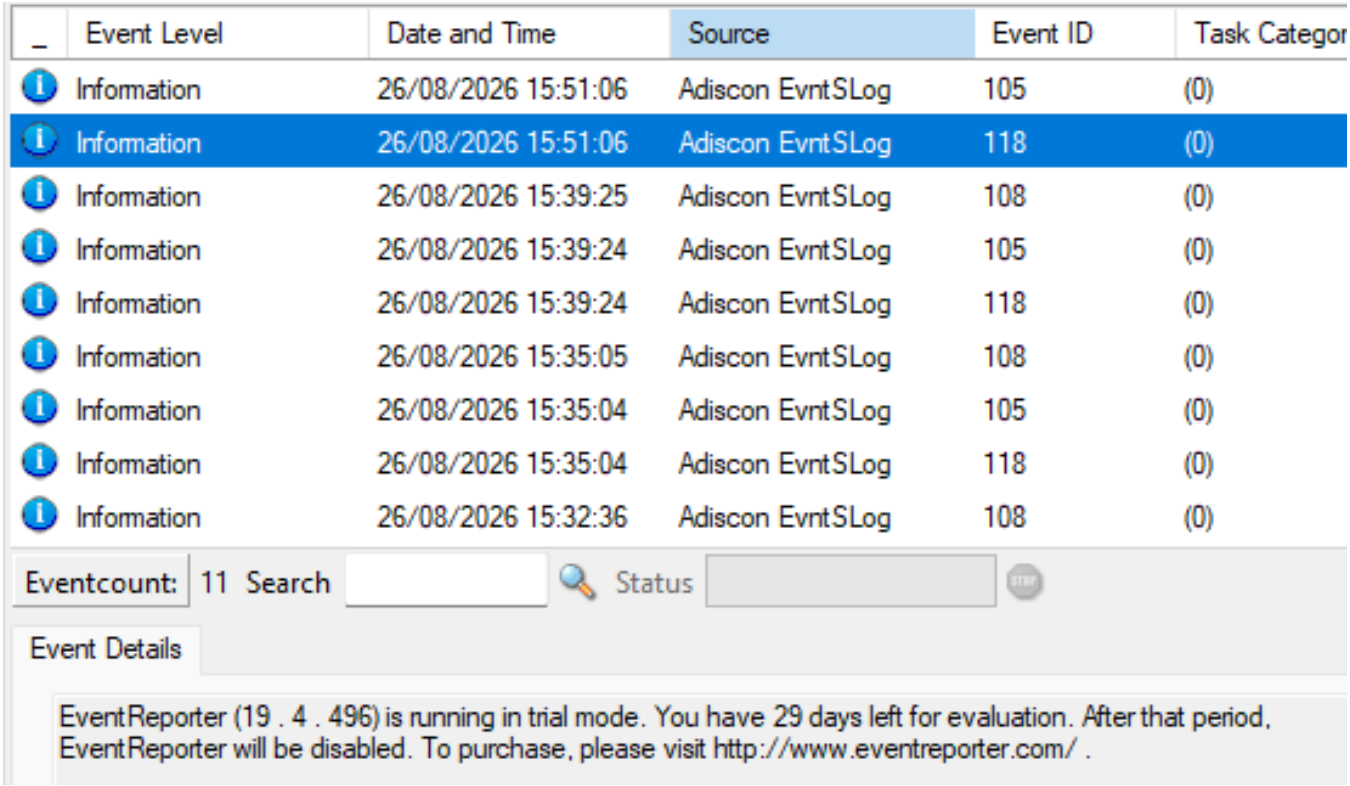

*A second maintenance run can reinstall the selected product files.*




*Files or services that remain in use can cause the installer to request a reboot.*




*After a successful file refresh, the client and service report the newer versions.*




*Verify the running product after the installer finishes.*

### Action path

1. Export or back up the WinSyslog configuration before changing the installation.

2. Close the WinSyslog Configuration Client and stop the WinSyslog service manually before each setup run. Stopping the service first can avoid a reboot request caused by locked service files, but it cannot avoid every reboot request.
3. If a staged update is practical, use a tested intermediate release, such as 16.x to 17.x to 18.x, and verify the client, service, features, and configuration after each hop. Do not assume that an adjacent version path is safe without verification.
4. If the installation remains inconsistent, use the clean-install procedure for WinSyslog after exporting the configuration.
5. If you are already working from the folder that contains the installer, run the following as two separate elevated setup runs. Substitute `wnsyslogjp.exe` when using the Japanese setup.

   First run: mark every setup feature as locally installed.

   ```
   wnsyslog.exe /v"ADDLOCAL=ALL /L*v %TEMP%\winsyslog-addlocal.log"
   ```

   Second run: force a refresh of the installed files and registration data.

   ```
   wnsyslog.exe /v"REINSTALL=ALL REINSTALLMODE=amus /L*v %TEMP%\winsyslog-reinstall.log"
   ```

   Let the wizard finish after each command. Do not combine the two property sets into one command for this recovery path. If the first command opens a resume page, continue through the wizard. After the second run, check Modify or Custom Setup, then verify the About dialog and the file versions before starting the service.

### Related information

- **`err-2026-002-winsyslog-cross-generation-installer-upgrade`**
- Installation
- Microsoft Windows Installer ADDLOCAL property
- Microsoft Windows Installer REINSTALL property
- Microsoft Windows Installer REINSTALLMODE property
- Microsoft Windows Installer system reboots

## Why do log files remain locked when multiple rules write to the same file?

This article explains why WinSyslog, EventReporter, and WinSyslog may continue to hold file handles to log files even after the configured timeout period, preventing external processes from accessing or archiving those files.

### Applies To

- WinSyslog
- EventReporter
- WinSyslog

### Problem

The service continues to hold file handles to log files even after the configured timeout period has elapsed. This prevents external processes such as batch scripts or archiving tools from accessing, moving, or archiving the log files.

### Symptoms

- Log files remain locked after the expected release time (e.g., after 2:00 AM daily when “Create unique filenames” is enabled)
- External batch scripts fail to archive log files because files are still in use by the service
- Files are not released even hours after the timeout period has passed
- CleanFileHandlesTimeout setting appears to be ignored or ineffective
- Error messages indicating files are in use when attempting to access them

### Root Cause

The issue occurs when multiple file actions use identical filename templates that can reference the same physical file. This creates a conflict in file handle management:

1. Each rule's file action creates its own independent file handle tracking mechanism
2. Multiple actions hold references to the same physical file simultaneously
3. The file cannot be released until ALL actions release their handles
4. Even though CleanFileHandlesTimeout is set correctly (e.g., 7200 seconds / 2 hours), the file remains locked because different actions have different timer states and may not all reach the timeout simultaneously

### Problematic Configuration Pattern

Multiple rules using identical filename templates:

- File Path: `log-directory\%timegenerated%`
- File Base Name: `%source%-%timegenerated%.log`

When multiple rules match messages from the same source device, they all write to the same physical file (e.g., `device-ip-20251012.log`). This causes handle conflicts where each action maintains its own handle to the same file.

### Example of the Issue

- Rule 1: Filter matches source "device-hostname" OR source "192.168.1.100" - File Action writes to `device-ip-20251012.log`
- Rule 2: Filter matches source "device-hostname" OR source "192.168.1.100" - File Action writes to `device-ip-20251012.log` (SAME FILE!)
- Rule 3-N: Same pattern with different filters but identical file action configuration

### Solution

#### Option 1: Consolidate Rules with Identical File Actions (Recommended)

Instead of having multiple separate rules with identical file action configurations, consolidate them into ONE rule with all filter conditions combined. This ensures only one file action object manages each file.

Steps:

1. Create a new rule (e.g., "All Devices Combined")
2. Use Copy & Paste functions in the configuration client to combine filters from all existing rules:
    - For each existing rule:
        - Navigate to the rule
        - Go to Filters tab
        - Find the filter group for that rule
        - RIGHT-CLICK on the filter block (usually an AND or OR block)
        - Select "Copy"
    - Navigate to your new "All Devices Combined" rule
    - Go to Filters tab
    - RIGHT-CLICK on the top-level OR filter
    - Select "Paste" (or "Paste as child")
    - The entire filter block is now copied into your new rule
    - Repeat this process for all existing rules
3. Configure the file action once in the new combined rule
4. Delete the other rules so that only a single File Action remains

**Result:** Only ONE file action object manages each file, eliminating handle conflicts completely.

#### Option 2: Use Unique File Paths for Each Rule

If you need to keep separate rules for organizational purposes, ensure each rule writes to a unique file location:

- Organize logs into separate subdirectories by device type, rule name, or other categorization
- Use different filename patterns for each rule
- Ensure no two rules can write to the same physical file under any circumstances

Example Configuration:

- Rule 1: File Path: `log-directory\devices\type1\%timegenerated%`
- Rule 2: File Path: `log-directory\devices\type2\%timegenerated%`
- Rule 3: File Path: `log-directory\devices\type3\%timegenerated%`

Each rule writes to a completely separate directory structure, preventing any possibility of file handle conflicts.

### Best Practices

- Each rule should write to a unique file location to avoid handle conflicts
- Consolidate rules with identical file action configurations into a single rule when possible
- Use the Copy & Paste functions in the configuration client to efficiently combine filter conditions from multiple rules
- When using “Create unique filenames” for daily rotation, ensure file paths are unique per rule
- Test file release behavior after configuration changes to verify files are released as expected
- Monitor file handles using Windows Resource Monitor or Process Explorer to verify proper release
- Verify that external batch scripts or archiving processes can access files after the timeout period

### Verification

After implementing the solution:

1. Monitor file handles using Windows Resource Monitor or Process Explorer to confirm files are released after the CleanFileHandlesTimeout period
2. Verify configuration using the configuration client’s verification feature - it should report 0 errors related to duplicate filenames

If files are still not released after the timeout period, check for:

- Other file actions or rules that may be holding references to the same files
- Additional processes that may be accessing the files
- Configuration errors that may have been missed during the consolidation process

## How to resolve performance issues on high-load systems?

This article explains how to resolve performance issues such as slow configuration reloads, extended service start/stop times, queue buildup, and potential timeouts during service operations on systems with high message volume.

### Applies To

- WinSyslog
- WinSyslog
- rsyslog Windows Agent

### Problem

On systems with high message volume, these products may experience performance issues such as slow configuration reloads, extended service start/stop times, queue buildup, and potential timeouts during service operations. These issues are more likely to occur when the system is processing many messages with insufficient worker threads.

### Symptoms

- Configuration reloads take longer than expected or appear to hang
- Service restart operations timeout with “Could not stop the service within 20 seconds” error

- Extended service start and stop times, especially when debug logging is enabled
- Queue buildup during high message volume periods
- Slow filter and action processing during peak load
- System appears unresponsive during configuration changes

### Root Cause

Under high message load, these products need sufficient parallel processing capacity to handle incoming messages efficiently. When worker threads are too few relative to the CPU cores and message volume:

- Filters and actions cannot run in parallel effectively
- Queues drain slowly, causing message buildup
- Configuration reload windows stretch because in-flight work drains slowly
- Service start/stop operations take longer because the service must wait for in-flight work to complete

The default worker thread count may be insufficient for high-load systems with multiple CPU cores.

### Solution

#### Primary Recommendation: Increase Worker Threads

The most important setting for high-load systems is the Worker Threads configuration:

**For WinSyslog:**

1. Open the WinSyslog Configuration Client
2. Navigate to **General Options > Queue Manager** section
3. Set **Number of worker threads** to at least **half the CPU core count**
    - Example: For an 8-core system, set to at least 4 worker threads
    - Example: For a 16-core system, set to at least 8 worker threads
4. Save the configuration
    - If automatic configuration reload is enabled, the service will reload automatically
    - If automatic configuration reload is disabled, manually restart the service

**For WinSyslog:**

1. Open the WinSyslog Configuration Client
2. Navigate to **General Options > Queue Manager** section
3. Set **Number of worker threads** to at least **half the CPU core count**
    - Example: For an 8-core system, set to at least 4 worker threads
    - Example: For a 16-core system, set to at least 8 worker threads
4. Save the configuration
    - If automatic configuration reload is enabled, the service will reload automatically
    - If automatic configuration reload is disabled, manually restart the service

**For rsyslog Windows Agent:**

1. Open the product configuration interface
2. Navigate to the **Queue Manager** or equivalent section (location may vary by product)
3. Set **Number of worker threads** to at least **half the CPU core count**
    - Example: For an 8-core system, set to at least 4 worker threads
    - Example: For a 16-core system, set to at least 8 worker threads
4. Save the configuration
    - If automatic configuration reload is enabled, the service will reload automatically
    - If automatic configuration reload is disabled, manually restart the service

**Why this helps:**

- Allows filters and actions to run in parallel

- Drains queues faster during high message volume
- Reduces configuration reload window duration under load
- Improves overall system responsiveness during peak periods

### Additional Recommendations for High-Load Systems

**Test Configuration Changes on Test System First**

For busy production systems, avoid making large configuration changes directly:

1. Perform larger configuration changes on a test system first
2. Verify the changes work correctly on the test system
3. Import the verified configuration into the production system
4. If automatic configuration reload is enabled, the service will reload automatically after saving
5. If automatic configuration reload is disabled, manually restart the service from the Configuration Client or Windows Services Management Console
6. Monitor system performance after the change

**Use Windows Services Management Console for Service Operations**

If automatic configuration reload is disabled and you need to restart the service, or if the configuration client service restart fails or times out:

1. Open Windows Services Management Console (services.msc)
2. Locate the product service (WinSyslog, WinSyslog, or rsyslog Windows Agent service)
3. Right-click and select **Restart** (or **Stop** then **Start**)
4. The Services Console provides more reliable service control under high load

**Note:** If automatic configuration reload is enabled, manual service restart is typically not needed after configuration changes.

**Allow Sufficient Time for Operations**

On high-load systems, allow adequate time for:

- Configuration reloads to complete
- Service restarts to finish
- Queue processing to catch up after changes

Avoid making multiple rapid configuration changes in succession.

### Best Practices

- **Set Worker Threads to at least half the CPU core count** - This is the most critical setting for high-load systems
- **Test configuration changes on a test system first** - Reduces risk of issues on production systems
- **Monitor system performance** - Watch for queue buildup, slow processing, or extended operation times
- **Avoid debug logging on production systems** - Debug logging significantly increases processing overhead and extends operation times
- **Plan configuration changes during lower-traffic periods** - When possible, schedule major changes during maintenance windows
- **Monitor Event ID 126 (WinSyslog)** - Verify that configuration reloads complete successfully (“Configuration reload successfully done”)

### Verification

To verify that worker thread settings are appropriate:

1. Check the current Worker Threads setting in QueueManager section
2. Determine your system’s CPU core count
3. Verify that Worker Threads is set to at least half the CPU core count
4. Monitor system performance during peak message volume:
    - Check for queue buildup

- Monitor service responsiveness
- Verify configuration reloads complete in reasonable time
- Check for timeout errors during service operations

If performance issues persist after adjusting worker threads, consider:

- Further increasing worker threads (up to the CPU core count)
- Reviewing filter and action complexity

## Is MariaDB supported by the ODBC action?

This article explains MariaDB support in ODBC database actions.

### Question

Is MariaDB supported by the ODBC action?

### Answer

Yes, MariaDB is fully supported by the ODBC action and can be used as a direct replacement for MySQL.

### Background

MariaDB is a free and open-source alternative to MySQL. It is a fork of MySQL, initiated by the original MySQL developers after Oracle acquired Sun Microsystems (the former owner of MySQL). MariaDB was designed to be binary-compatible with MySQL, which generally makes switching from MySQL to MariaDB very easy.

Key characteristics of MariaDB:

- **Open Source:** MariaDB is consistently Open Source under a license that guarantees free use and further development
- **Binary Compatibility:** Designed to be binary-compatible with MySQL, making migration straightforward
- **Independent Development:** Continuous, independent development separate from MySQL
- **Performance:** Often preferred as an alternative due to sometimes better performance characteristics

### Configuration

To use MariaDB with the ODBC action:

1. **Install MariaDB ODBC Driver:** - Download and install the MariaDB Connector/ODBC driver from the official MariaDB website - Ensure you install the correct version (32-bit or 64-bit) to match your Adiscon product installation
2. **Configure System DSN:** - Open the ODBC Data Source Administrator (use the 32-bit version if your product runs in 32-bit mode) - Create a new System DSN - Select the MariaDB ODBC driver - Configure the connection settings (server, database, credentials)
3. **Configure Database Action:** - In your Adiscon product configuration, select the ODBC Database action - Choose the MariaDB System DSN you created - Test the connection using the “Verify Database” button - Create the database tables if needed using the “Create Database” button

**Note:** The configuration process is identical to configuring MySQL, as MariaDB uses MySQL-compatible drivers and protocols.

### Modern Deployment Recommendations

For current MariaDB deployments, we recommend the following:

1. **Use a current MariaDB server and connector** - Use currently supported MariaDB server releases - Use a current MariaDB Connector/ODBC package from the official source
2. **Use a dedicated database account** - Create a dedicated user for the Adiscon product - Grant only the required privileges on the target database/schema
3. **Enable secure transport for remote database connections** - Use TLS between the Adiscon host and MariaDB server when traffic crosses networks - Configure certificate settings in the DSN/driver according to your security policy

4. **Use UTF-8 consistently** - Prefer UTF-8/`utf8mb4` settings for server, database, and connector - This prevents character conversion issues in international log messages
5. **Validate end-to-end before production rollout** - Use “Verify Database” in the ODBC action - Insert sample messages and verify they are written and readable as expected

### Common Modern Troubleshooting Checks

If connection tests fail, verify:

- Driver architecture matches the product runtime (32-bit vs 64-bit)
- Host, port, database name, and credentials in the DSN are correct
- MariaDB user authentication method is supported by the installed connector
- TLS requirements (if enabled) match server and connector configuration
- Firewall rules allow database traffic

### Additional Information

For more information about database actions, see the ODBC Database Options documentation in your product’s manual.

For MariaDB-specific information, visit the official MariaDB website.

## Recommended Palo Alto Firewall Syslog Configuration

This article provides configuration recommendations for Palo Alto firewalls to ensure consistent and reliable syslog message parsing by your syslog server.

### Question

What is the recommended syslog format configuration for Palo Alto firewalls when sending logs to a syslog server?

### Answer

**We recommend configuring Palo Alto firewalls to use IETF** RFC 5424 **syslog format instead of BSD format.** The IETF format provides a structured, unambiguous message format that ensures consistent parsing regardless of Palo Alto firmware version or spacing differences in log messages.

### Why Use IETF (RFC 5424) Format?

IETF format is recommended over BSD

format for the following reasons:

1. **Structured format:** IETF format includes a required APP-NAME field that eliminates parsing ambiguity
2. **Consistent parsing:** The structured format ensures your syslog server parses messages consistently regardless of: * Palo Alto firmware version * Spacing differences in log messages * Future firmware updates that may change message formatting
3. **Better compatibility:** IETF format is the modern syslog standard and provides better support for SIEM systems and log analysis tools
4. **Prevents parsing issues:** BSD format relies on heuristics that can be affected by spacing changes, potentially causing fields like `version=` to be parsed incorrectly or missing from output

**Note:** If you’re experiencing issues where the `version=` field is missing from syslog output after a Palo Alto upgrade, this is typically caused by BSD format parsing ambiguity due to spacing changes. Switching to IETF format resolves this issue.

### Configuration Steps

#### Step 1: Access Syslog Server Profile

1. Log in to the Palo Alto Networks firewall web interface

2. Navigate to: **Device > Server Profiles > Syslog**

   Reference: Palo Alto Documentation - Configure Syslog Monitoring
3. Either: * Edit an existing syslog server profile, or * Click **Add** to create a new profile

### Step 2: Configure Syslog Server Settings

For each syslog server in the profile:

1. **Name:** Enter a unique name for the server (if creating new)
2. **Syslog Server:** Enter the IP address or FQDN of your syslog server
3. **Transport:** * **Important:** IETF format typically uses TCP or SSL (TLS) * Select **TCP** or **SSL** (not UDP) * If using SSL, ensure TLSv1.2 is supported

   Reference: Palo Alto Documentation - Configure Syslog Monitoring
4. **Port:** Enter the port number (default TCP syslog port is 514, but verify with your syslog server configuration)
5. **Format:** Select **IETF** (this is the key setting)

   Reference: Palo Alto Documentation - Configure Syslog Monitoring
6. **Facility:** Select the appropriate syslog facility value (default is LOG_USER)

### Step 3: Verify The Syslog Service Supports RFC 5424

Before applying the changes, ensure:

1. The Syslog Service supports RFC 5424 format: Verify that RFC 5424 parsing is enabled

### Step 4: Commit Configuration

1. Click **OK** to save the syslog server profile
2. Commit the configuration
3. Review the commit and click **Commit** again to confirm

   Reference: Palo Alto Documentation - Configure Syslog Monitoring

### Step 5: Verify Configuration

After committing:

1. Check syslog messages on your syslog server
2. Verify the format: Messages should now appear in IETF format:

   ```
   <14>1 2025-10-30T13:13:04.000Z e26secgw02 paloalto - - [meta version="11.2.6"] version=11.2
   .6|subtype=general|...
   ```
3. Verify APP-NAME field: The `paloalto` field (APP-NAME) should be present and consistently parsed by your syslog server
4. Verify output format: Syslog server output should now consistently include the `version=` prefix

## Expected Results

After configuring IETF format, you should see:

- **Consistent message format:** Messages appear in structured IETF format with the APP-NAME field (`paloalto`) consistently parsed
- **Reliable field extraction:** All fields, including `version=`, are reliably extracted regardless of Palo Alto firmware version
- **Future-proof configuration:** The structured format ensures consistent behavior even after firmware upgrades
- **Better log analysis:** The structured format provides better support for SIEM systems and log analysis tools

## Benefits Summary

Using IETF (RFC 5424) format provides:

- **Eliminates parsing ambiguity:** The structured format with required APP-NAME field ensures consistent parsing
- **Prevents version-related issues:** Spacing changes in firmware updates won't affect message parsing
- **Industry standard:** IETF format is the modern syslog standard recommended for enterprise environments
- **Better integration:** Improved compatibility with SIEM systems, log analysis tools, and centralized logging solutions

#### Technical Reference

- RFC 5424 (IETF)
- Palo Alto Documentation - Configure Syslog Monitoring
- Palo Alto Documentation - Syslog Field Descriptions
- Palo Alto Documentation - Use Syslog for Monitoring

#### Additional Information

For more information about syslog server configuration and RFC 5424 support, see the Syslog Server documentation in your product's manual.

## Deployment and administration

### How Do I Perform a Repeatable Deployment of WinSyslog?

#### Overview

This FAQ explains how to deploy WinSyslog to multiple machines by reusing one validated reference configuration. This approach is useful whenever more than one system should receive the same or similar WinSyslog setup.

#### When to Use Repeatable Deployment

A repeatable deployment is appropriate when:

- more than one machine should use the same baseline configuration
- you want to automate distribution across multiple systems
- you are setting up remote offices with consistent configurations
- you want to standardize syslog infrastructure across an organization

**Note:** For a single one-off install, manual setup is usually simpler. The repeatable method becomes useful as soon as you want to reuse the same package and configuration across multiple systems.

#### Approach Overview

The repeatable deployment process involves:

1. Creating a reference configuration on one system
2. Exporting the configuration to a registry file
3. Distributing the registry file and service executable to target systems
4. Importing the configuration and starting the service

#### Step-by-Step Repeatable Deployment

##### Step 1: Create Reference Configuration

1. Install WinSyslog on a reference system
2. Configure all rules, services, and actions as desired
3. Test the configuration thoroughly
4. Export the configuration using the Configuration Client

#### Step 2: Export Configuration

1. Open the WinSyslog Configuration Client
2. Go to “Computer” menu
3. Select “Export Settings to Registry File”
4. Specify a filename (e.g., `winsyslog-config.reg`)
5. Save the file

#### Step 3: Prepare for Distribution

You need to distribute:

- **Service executable:** `winsyslg.exe` (from installation directory)
- **Helper DLLs:** Required DLL files
- **Configuration file:** The exported `.reg` file

#### Step 4: Automated Installation Script

Create a batch file to automate the installation:

```
@echo off
REM Repeatable Deployment Script for WinSyslog
REM This script installs and configures WinSyslog on target machines

REM Copy service executable from network share
copy \\servershare\winsyslg.exe c:\winsyslog\
copy \\servershare\*.dll c:\winsyslog\

REM Navigate to installation directory
cd c:\winsyslog

REM Register the service (creates registry entries)
winsyslg.exe -i

REM Import the configuration
regedit /s \\servershare\winsyslog-config.reg

REM Start the service
net start "AdisconWINSyslog"

echo WinSyslog installed and started successfully
```

### Engine-Only Installation

For locked-down environments or when you don’t want to run the full setup program:

- **No Windows RPC required:** Engine-only install doesn’t require incoming RPC connections
- **Minimal footprint:** Only essential service files needed
- **Direct execution:** Files can be copied and service registered directly

**Important:** The `winsyslg.exe -i` command does NOT copy files. It assumes they are already in their final location and only creates the necessary registry entries to register WinSyslog as a Windows service.

### Branch Office Deployment

For remote offices where some manual intervention is acceptable:

#### Preparation (Central Office)

1. Complete installation on one machine
2. Configure settings as desired
3. Export configuration to `.reg` file

4. Copy the following files to a CD or network share:
    - `winsyslg.exe`
    - `winsyslg.pem`
    - Required DLL files
    - Configuration `.reg` file

### Installation (Branch Office)

1. Create installation directory (e.g., `C:\Program Files\WinSyslog`)
2. Copy all files from CD to the directory
3. Run `winsyslg.exe -i` from that directory
4. Double-click the `.reg` file to import configuration
5. Start the service using Services Manager or `net start "AdisconWINSyslog"`

### Modern Deployment Alternatives

For contemporary environments, consider these options:

- **Group Policy:** Use GPO to deploy registry files and execute installation scripts
- **PowerShell DSC:** Desired State Configuration for automated deployment
- **Configuration Management:** Integrate with SCCM, Ansible, or similar tools
- **Cloud Deployment:** Automate via cloud provisioning scripts for virtual machines

### Best Practices

- **Test first:** Always test deployment on a non-production system
- **Document changes:** Keep track of configuration modifications
- **Backup existing:** Backup any existing configuration before deployment
- **Verify service:** Check that service starts correctly after deployment
- **Monitor logs:** Review event logs for any installation issues
- **Update paths:** Ensure file paths are appropriate for target environment

### Troubleshooting

Common issues and solutions:

**Service won't start**
: Verify executable permissions and file paths

**Configuration not applied**
: Check registry import was successful

**Missing DLLs**
: Ensure all required DLL files are copied

**Port conflicts**
: Verify syslog ports aren't already in use

**Firewall issues**
: Ensure Windows Firewall rules are configured

### Additional Notes

- The installation directory becomes the permanent program directory – do not delete it
- All configuration is stored in the Windows registry
- Service name is `AdisconWINSyslog` (internal name)
- Consider using file-based configuration (`.cfg` files) for Server Core deployments
- For high-availability, plan for service dependencies and startup order

### See Also

- Installation - WinSyslog installation documentation

## Why did my WinSyslog update report success when the files were not upgraded correctly?

### Question

Why can an in-place WinSyslog update report success while the installed service, client, or selected features still contain files from the older release?

### Answer

This can happen when Windows Installer retains an incomplete feature state while processing a cross-generation update. The setup wizard can finish and report success even though a later maintenance operation still sees features as advertised, unavailable, or otherwise not fully installed. The product configuration is normally not deleted; the problem is that the expected program files were not refreshed.

### Details

This behavior has been confirmed for direct WinSyslog updates from the 16.x generation to the 18.x generation. Other direct source versions can show the same symptoms, including paths within the adjacent release generations. The source version alone does not prove that an update path is safe or affected.

The underlying mechanism is consistent with Windows Installer feature-state handling. A newer setup package can be accepted as an update without changing the installed product identity, but Windows Installer still evaluates the feature and component state recorded for the previous installation. A change between product generations can leave a feature only partially selected or advertised. `ADDLOCAL=ALL` explicitly requests all features locally, while `REINSTALL=ALL` only refreshes features that Windows Installer considers installed. This is why a two-stage repair is useful for this symptom.

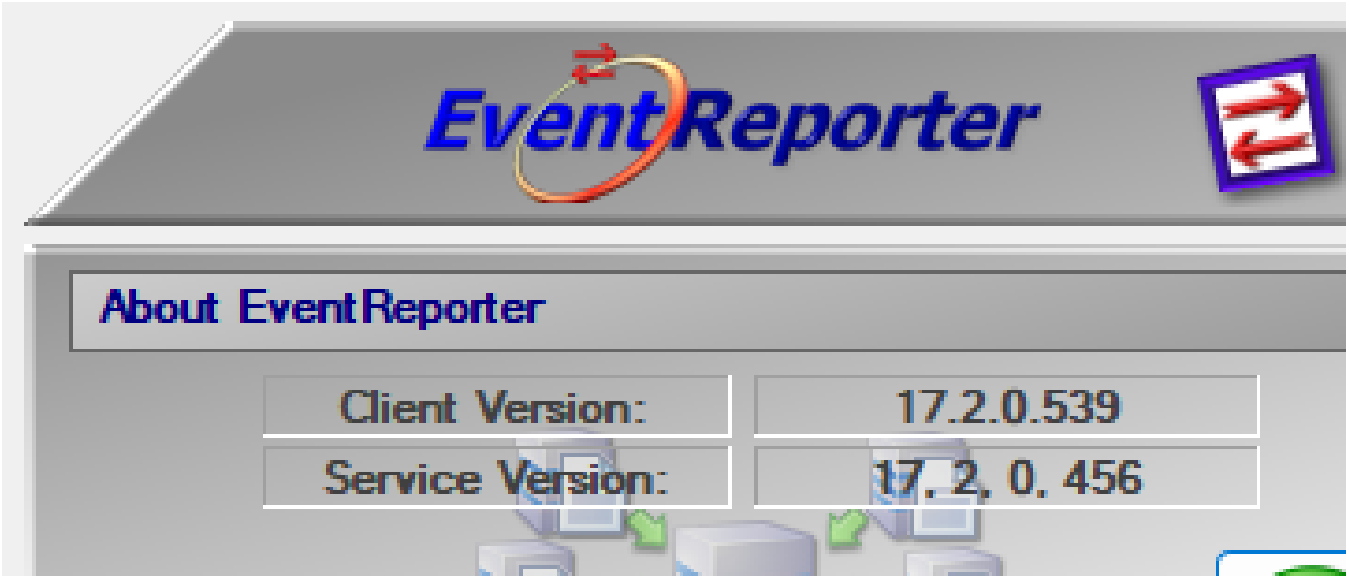
Look for this combination:

- the setup wizard displays a successful completion page
- the WinSyslog About dialog or file properties still show an older client or service version
- Modify or Custom Setup shows an unexpected state such as `0 of 1 subfeatures selected` or `0 KB`
- a normal repair does not replace the older files
- the installer requests a reboot because a service or file is still in use

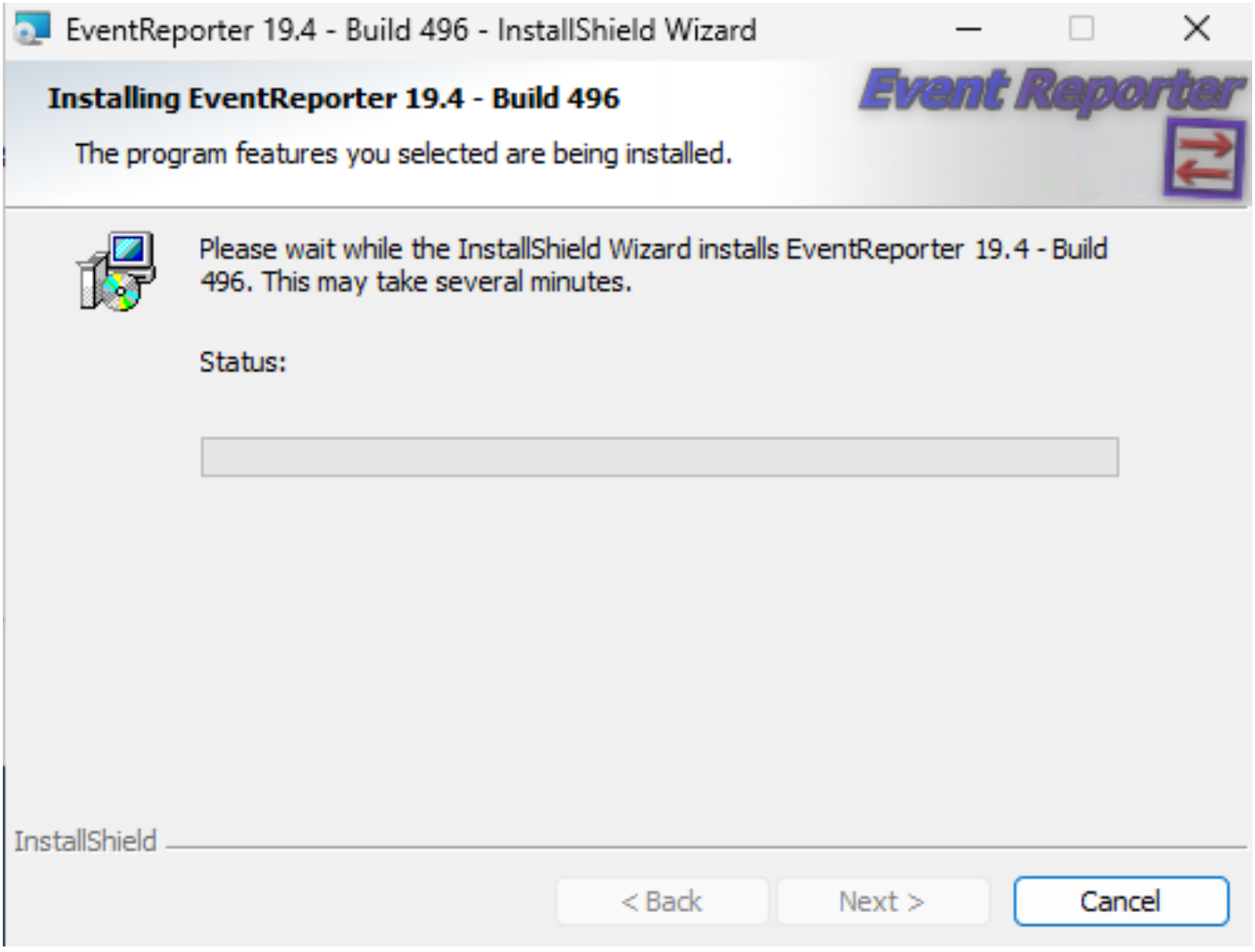
**orphan:**

Representative setup evidence

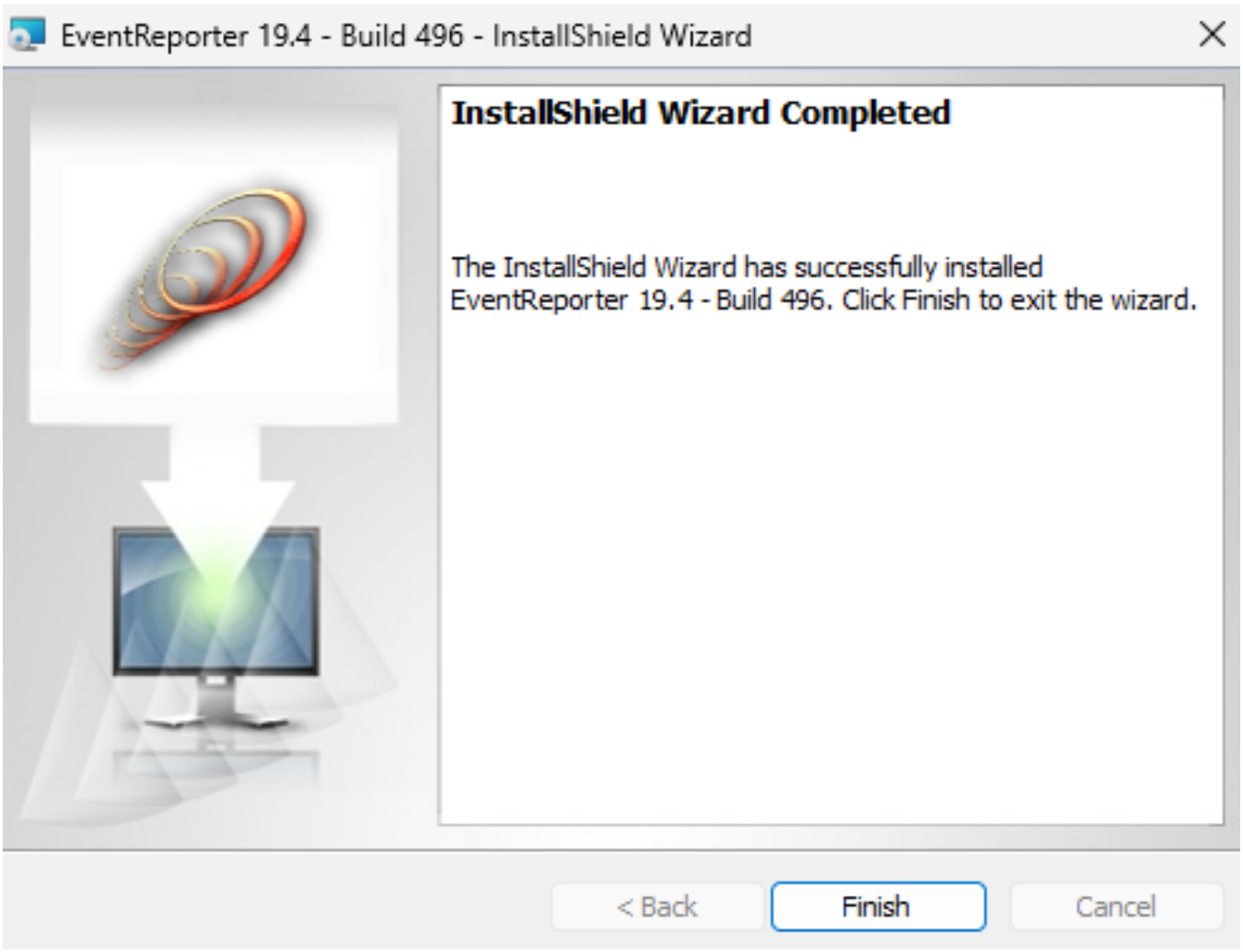
The screenshots below come from one affected setup run and are representative of the visual clues described in this article. Product names, component labels, and version numbers can differ in other product installations.



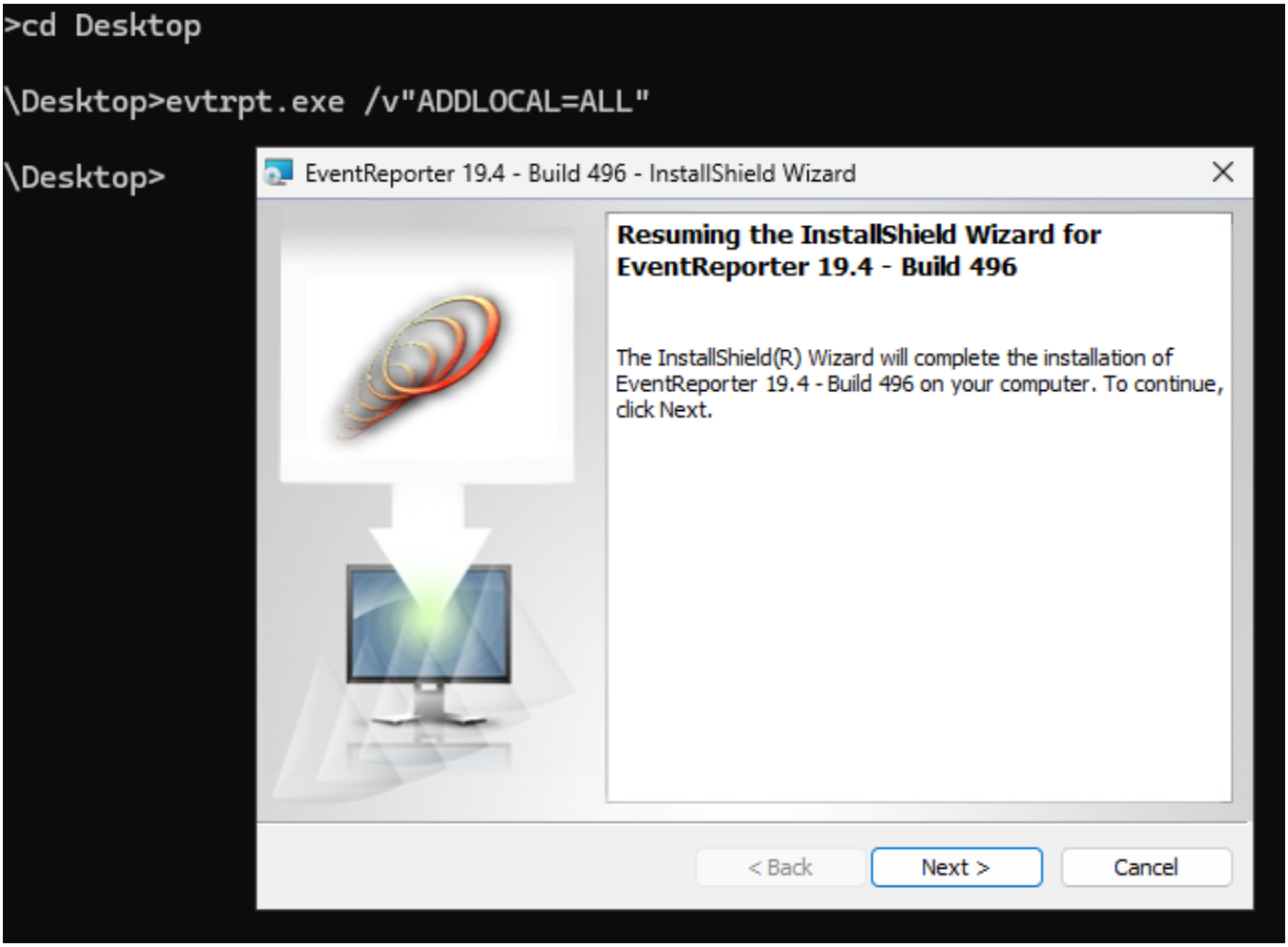

*The installed product reports the older client and service versions before the update.*



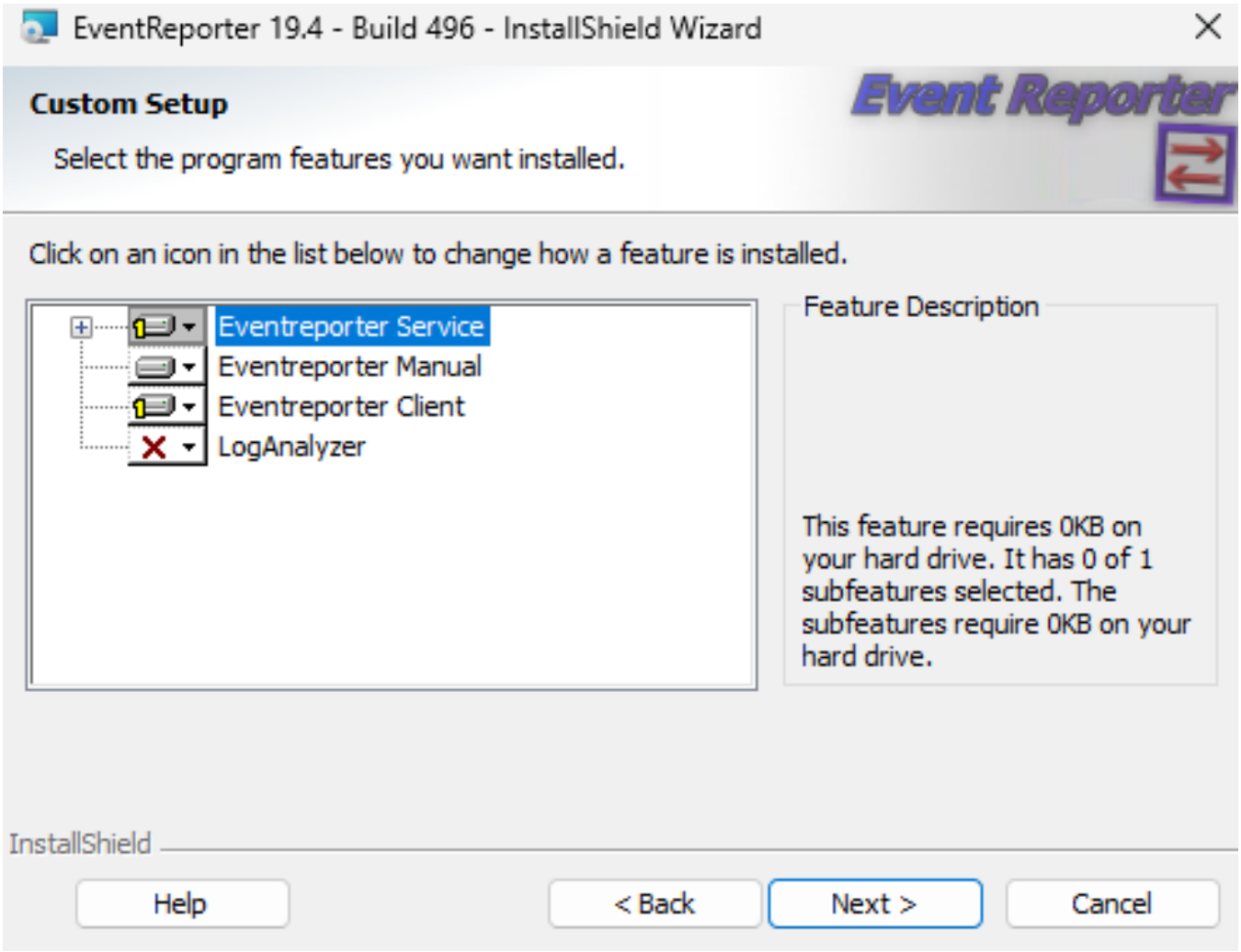

*The setup wizard can display a normal installation-progress page.*



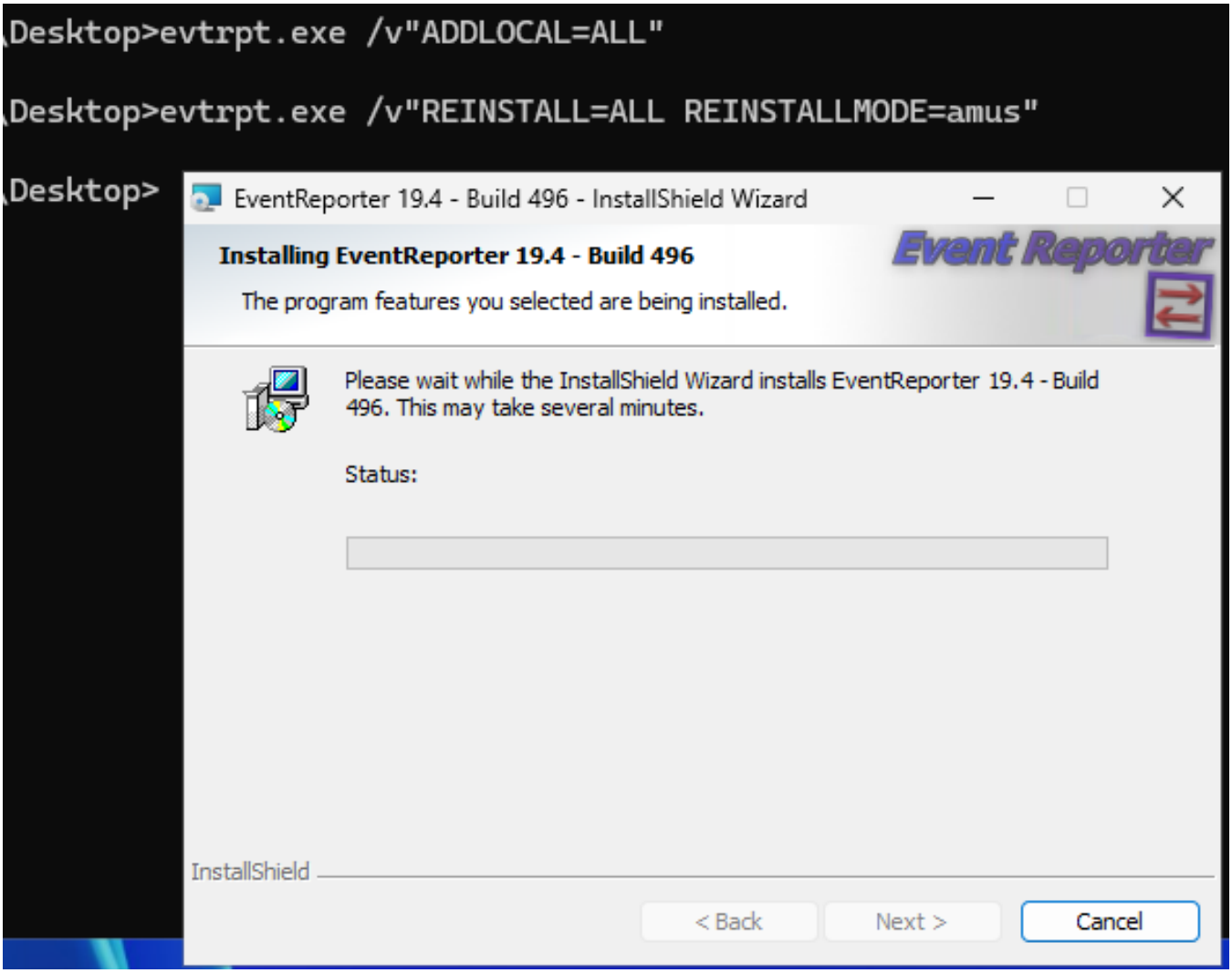

*The wizard can report that the installation completed successfully.*



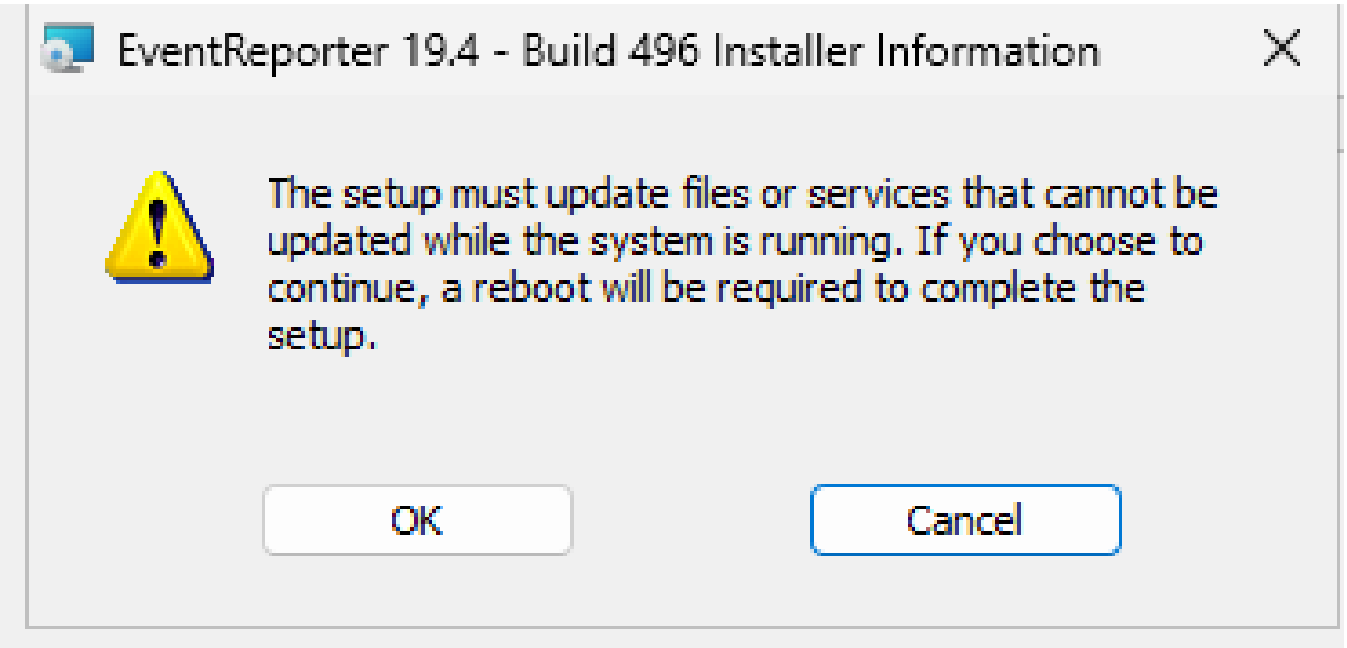

*A later maintenance action can resume the InstallShield wizard.*




*Custom Setup can show an unexpected feature state, such as zero selected subfeatures or zero required disk space.*



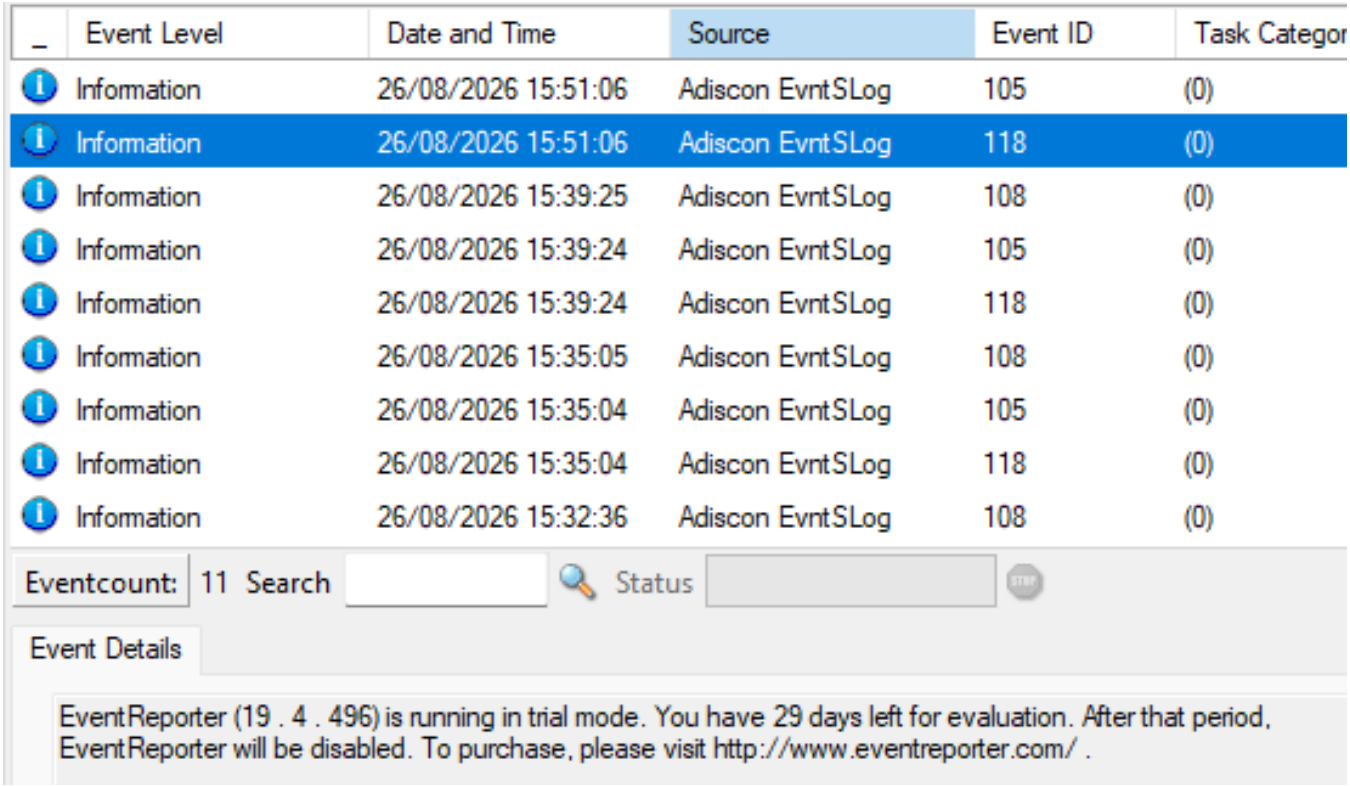

*A second maintenance run can reinstall the selected product files.*




*Files or services that remain in use can cause the installer to request a reboot.*




*After a successful file refresh, the client and service report the newer versions.*




*Verify the running product after the installer finishes.*

### Action path

1. Export or back up the WinSyslog configuration before changing the installation.

2. Close the WinSyslog Configuration Client and stop the WinSyslog service manually before each setup run. Stopping the service first can avoid a reboot request caused by locked service files, but it cannot avoid every reboot request.
3. If a staged update is practical, use a tested intermediate release, such as 16.x to 17.x to 18.x, and verify the client, service, features, and configuration after each hop. Do not assume that an adjacent version path is safe without verification.
4. If the installation remains inconsistent, use the clean-install procedure for WinSyslog after exporting the configuration.
5. If you are already working from the folder that contains the installer, run the following as two separate elevated setup runs. Substitute `wnsyslogjp.exe` when using the Japanese setup.

   First run: mark every setup feature as locally installed.

   ```
   wnsyslog.exe /v"ADDLOCAL=ALL /L*v %TEMP%\winsyslog-addlocal.log"
   ```

   Second run: force a refresh of the installed files and registration data.

   ```
   wnsyslog.exe /v"REINSTALL=ALL REINSTALLMODE=amus /L*v %TEMP%\winsyslog-reinstall.log"
   ```

   Let the wizard finish after each command. Do not combine the two property sets into one command for this recovery path. If the first command opens a resume page, continue through the wizard. After the second run, check Modify or Custom Setup, then verify the About dialog and the file versions before starting the service.

### Related information

- **`err-2026-002-winsyslog-cross-generation-installer-upgrade`**
- Installation
- Microsoft Windows Installer ADDLOCAL property
- Microsoft Windows Installer REINSTALL property
- Microsoft Windows Installer REINSTALLMODE property
- Microsoft Windows Installer system reboots

## How to Copy the WinSyslog Configuration to Other Servers

### Overview

If you have multiple copies of WinSyslog and want to replicate the current configuration across several servers, you can export the settings to a registry file and distribute it. This saves time by avoiding manual reconfiguration of services and rulesets on each machine.

### Exporting the Configuration

1. Open the WinSyslog Configuration Client.
2. Go to the **Computer** menu and click **Export Settings to Registry File** (not binary).
3. Save the registry file to a convenient location, for example a shared network folder.

### Importing the Configuration on Other Servers

On each target server, double-click the exported registry file. This imports all WinSyslog registry settings automatically. When you open the WinSyslog Configuration Client on that machine, your configured rules and services will be present.

### See Also

- How Do I Perform a Repeatable Deployment of WinSyslog? - How to perform a repeatable deployment of WinSyslog
- How to Export WinSyslog Settings for a Support Call - How to export WinSyslog settings for a support call

## How Do Remote Administration and Browser-Based Log Review Fit Together?

### Question

How do remote administration and browser-based log review work in the current Adiscon Windows products?

### Answer

Treat them as two separate functions.

- Administrative changes are made with the Configuration Client, either on the local system or through the remote-connect workflow when that product supports it.
- The current product family does not use a built-in browser administration interface for service configuration.
- Browser-based review is handled by Adiscon LogAnalyzer, which is a separate and optional component for stored data. It is not the service administration interface.

### Details

#### Remote administration

The current product family uses the Windows Configuration Client for administration. Depending on the product and deployment style, you typically work in one of these ways:

- connect to another machine from the client
- export and import configuration
- use a repeatable deployment process for repeated deployments

This means remote administration is a client-and-deployment workflow, not a built-in browser administration console.

When the client connects to another machine directly, the target system must be reachable over the network and the current user must have the required access rights on that remote machine.

#### Browser-based log review

Adiscon LogAnalyzer is the browser-based component in this ecosystem. It is used to review data that has already been written to a file or database. It is deployed separately from the logging service and requires its own web server and PHP runtime.

That split is important:

- LogAnalyzer is for stored data review.
- The logging service still receives, processes, and stores or forwards data.
- The Configuration Client is still the place where you change service settings.

#### What this means operationally

If you want both remote administration and browser-based visibility, plan them as separate pieces:

1. Decide how the product configuration will be administered remotely.
2. Decide where retained data will be stored.
3. Deploy LogAnalyzer separately if browser-based review of stored data is needed.

### Action path

1. Use the Configuration Client for administrative changes.
2. If the target system is remote, use the product's remote-connect or deployment workflow.
3. Configure file or database storage for the data you want to review later.
4. Deploy Adiscon LogAnalyzer separately on a web server if browser-based review is required.
5. Point LogAnalyzer at the same stored data source and verify end-to-end that new rows appear there.

### Related information

- `../tutorials/loganalyzer-setup-and-use`
- Use the product-specific remote-connect page when the Configuration Client can connect to another machine directly.
- Use the product-specific deployment or configuration-copy guidance when you need repeatable deployment across multiple systems.

## How Can I Obtain a Printable Version of the WinSyslog Manual?

### Answer

You can use the online WinSyslog manual in your browser or download a printable PDF version from the WinSyslog manual site:

WinSyslog Manual

### Details

The online manual is usually the best choice for browsing, searching, and following links between related topics. If you need a printable copy for offline review, internal distribution, or change control, use the PDF version available on the manual site.

If WinSyslog is already installed, local help is also included with the installation, so an additional download is often unnecessary.

The web-hosted manual may be newer than the locally installed help because documentation updates are published online as they become available.

### Related Information

- Installation

## Can WinSyslog Write to a UNC Path?

### Question

Can WinSyslog write log files to a UNC path such as `\server\share\logs`?

### Answer

Yes, but not with the default service account. WinSyslog runs as a Windows service, and the default Local System account normally cannot access network shares. To use a UNC path, run the WinSyslog service under an account that has permission to access the target share.

### Details

UNC path support itself is not the problem. The limiting factor is the Windows security context under which the service runs.

Typical working setup:

- a domain or local service account is configured for the WinSyslog service
- that account has write permissions on the target share
- the share and underlying NTFS permissions both allow access

Typical non-working setup:

- the WinSyslog service still runs as Local System
- the target is a remote UNC share
- no explicit network permissions exist for the service account

### Action path

1. Create or choose a service account that has access to the target share.
2. Grant that account the required share and NTFS permissions.
3. Open the Windows Services console.
4. Open the properties of the `AdisconWINSyslog` service.
5. Change the logon account from Local System to the chosen service account.
6. Restart the WinSyslog service.
7. Configure the file action to use the UNC path.
8. Send a test message and verify that the target file is created or updated.

### Important notes

- Test the target path with the same account context that the service uses.
- Drive-letter mappings created in an interactive user session are usually not visible to Windows services. Use the UNC path directly.
- If possible, keep log storage local and use forwarding or downstream syncing for remote distribution. This is usually more robust than writing directly to a network share.

### Related information

- File Logging Options
- Tutorial: Create a Simple Syslog Server and Write to a File

## Which Database Format Should I Use with WinSyslog?

### Question

Which database format should I use with WinSyslog, and what should I consider before logging messages into a database?

### Answer

Use the default WinSyslog database format when you want the fastest supported setup or compatibility with the standard Adiscon table layout. Use a custom schema when WinSyslog must integrate with an existing database design.

If you want a hands-on Microsoft SQL Server example that you can reproduce immediately, start with Quick Start: Write Syslog Messages to Microsoft SQL Server.

WinSyslog is not limited to an “internal schema” path. The database action can write to supported ODBC-accessible databases with user-defined schemas, as long as you configure the table name and field mapping correctly.

For most production deployments, use a server-grade database such as Microsoft SQL Server, MySQL, MariaDB, or PostgreSQL. Avoid Microsoft Access for production logging.

### Details

WinSyslog can write messages into an ODBC-accessible database through the Write to Database action. The built-in default format is the safest choice for first-time setup because it matches the fields that Adiscon tools expect and avoids unnecessary mapping work. However, the action is also a general integration feature: it can write to your own table and column layout when that is the real requirement.

Use the default format when:

- you are setting up database logging for the first time
- you want predictable field mapping
- you plan to use the built-in **Create Database** flow
- you plan to analyze the data with Adiscon-compatible or standard SQL tooling
- you do not have an existing schema that WinSyslog must integrate with

Use a custom format only when:

- your organization already has a fixed schema

- another system requires specific column names or data types
- you have tested the mapping and understand the compatibility impact

#### What the database action does not do

The action is a writer and mapping layer. It does not design your schema, choose your indexes, or build your reporting model for you. If you choose a custom schema, you own the destination database design.

#### Production recommendation

For production use, prefer a database engine designed for concurrent writes and multi-user access. Microsoft Access is useful only for very small, temporary, or lab-style setups. It is prone to locking conflicts and does not scale well for continuous log ingestion.

This is especially important when multiple tools or users access the database at the same time. File-based database engines are much more likely to produce “file already in use” or similar locking problems.

#### Action path

1. Decide whether you need the default supported schema or integration with an existing custom schema.
2. Create and test an ODBC **System DSN** on the WinSyslog host.
3. If the DSN uses Windows authentication, remember that WinSyslog normally runs under the default Windows `Local System` service account unless you changed it. For a remote SQL Server, either grant SQL access to that service context, change the service logon account, or use SQL authentication.
4. In WinSyslog, add a Write to Database action to the ruleset that should store messages.
5. For the default schema path, follow Quick Start: Write Syslog Messages to Microsoft SQL Server.
6. For the custom integration path, follow Tutorial: Integrate WinSyslog with a Custom Database Schema.
7. Send a test message and verify that a row is inserted into the intended table.

#### Related information

- ODBC Database Options
- Quick Start: Write Syslog Messages to Microsoft SQL Server
- Tutorial: Integrate WinSyslog with a Custom Database Schema
- mariadb-odbc-support

## Protocols and integration

### What do “service”, “input”, “listener”, and “server” mean in WinSyslog?

#### Question

In the WinSyslog documentation and GUI, what is the difference between a `service`, an `input`, a `listener`, and a `server`?

#### Answer

In WinSyslog, **service** is the main documentation term, and **input** is the plain-language concept it represents.

- A **service** is a configured WinSyslog input that receives or generates events and sends them into a ruleset.
- An **input** is the easiest plain-language way to think about a receive-side service.
- A **listener** is the network-facing part of a service when the discussion is about IP address, port, transport, or TLS-related listen settings.
- A **server** usually appears because it is part of an existing GUI label, such as `Syslog server` or `SETP Server`, or because WinSyslog is being described at the product level as a Windows syslog server.

If you want the simplest reading: think “add an input service that receives logs”, even when the GUI page is named `Syslog server` or `RELP Listener`.

### Details

WinSyslog uses one Windows service process, but inside the product you can configure multiple **services**. Each configured service has its own settings and its own associated ruleset.

Some of those services are network inputs:

- `Syslog server` service receives syslog.
- `RELP Listener` service receives RELP.
- `SETP Server` service receives SETP.
- `SNMP Trap Receiver` service receives SNMP traps.

The GUI labels are technical and must stay exact in the documentation where users need to match what they see on screen. That is why this manual keeps names like `Listener Port` and `Syslog server` in field descriptions and step-by-step UI instructions.

However, using only those labels can be hard for new users and for AI systems that read the generated HTML. This manual therefore uses the following pattern:

- Use **service** as the default concept in prose.
- Read **service** as “configured input” when you want the clearest plain language.
- Use the exact GUI name the first time it matters, for example `Syslog server` service.
- Use **listener** only when the topic is network listen behavior, such as port conflicts, IP address selection, or TLS coexistence.
- Use **receiver** only for sender/receiver workflows where two systems are communicating.

This keeps the GUI terminology accurate while still making the operational role easy to understand.

### Action path

1. When you read a setup page, treat **service** as the main WinSyslog input concept and **input** as the plain-language meaning of that service.
2. When the GUI shows a specific label such as `Syslog server` or `RELP Listener`, match that label exactly in the client.
3. When a page discusses port, IP address, or transport conflicts, read **listener** as “the network listen behavior of that service”.
4. When a page discusses forwarding between two systems, read **sender** and **receiver** as the remote communication roles, not as different WinSyslog object types.

### Related information

- Services
- How Do Port, Address, and Transport Conflicts Work for Input Services?

## How Do Port, Address, and Transport Conflicts Work for Input Services?

### Question

When I create multiple input services, which combinations can run at the same time and which ones conflict?

### Answer

Treat each receive-side service as an input service. When a GUI field or older page says **listener**, it refers to the network side of that input service. In practice, the relevant settings are the transport protocol, the local IP address, and the local port.

Two input services can run side by side only when those settings do not conflict. Changing the transport, IP address, or port avoids the conflict. If two input services need the same combination, only one of them can use it.

TLS does not change that rule. A TLS-enabled input service still binds a TCP port, so plain TCP and TCP+TLS cannot both listen on the same IP address and port.

### Details

Use this rule of thumb:

- Different transport protocols can coexist on the same port, for example UDP/514 and TCP/514.
- TCP and TCP+TLS cannot both use the same IP address and port, because both bind TCP at the socket level.
- Different ports can coexist, for example TCP/514 and TCP+TLS/1514.
- Different local IP addresses can also coexist, as long as the input type exposes that setting.

For most administrators, the simplest practical rule is this:

- Reusing the same port number is usually safe only when the transport differs, for example UDP versus TCP.
- Do not expect reuse to work when both services ultimately use TCP on the same local address, even if the higher-level protocol is different.
- TLS does not create a separate transport; it still runs on top of TCP.
- DTLS is a notable exception because it runs over UDP.

### Short IP primer

The special address `0.0.0.0` means “all local IPv4 addresses” and `::` means “all local IPv6 addresses”. If one input service already uses `0.0.0.0` on a given transport and port, another input service with the same transport usually cannot reuse that port on one specific IPv4 address because the wildcard address already covers it.

### GUI-specific differences

The exact UI fields depend on the input type:

- Some input services expose transport, IP address, and port directly.
- Some input services expose only part of that combination.
- Some input services support only one transport mode and therefore do not offer a transport choice at all.
- Some input services separate IPv4 and IPv6 into different services or settings.

The conflict rule still stays the same: one active input service per effective address, port, and transport combination.

In WinSyslog-oriented wording, that usually means one active network input service per effective address, port, and transport combination.

### Concrete examples

These combinations are valid:

- UDP / `0.0.0.0` / `514`
- TCP / `0.0.0.0` / `514`
- TCP+TLS / `0.0.0.0` / `1514`

This combination conflicts:

- TCP / `0.0.0.0` / `514`
- TCP+TLS / `0.0.0.0` / `514`

### Action path

1. Identify the transport, local IP address, and local port required by each input service.
2. Check whether any active input service already uses the same effective address, port, and transport combination.
3. If there is a conflict, change the port or bind to a different local IP address when that input type allows it.
4. If TLS is required in parallel with plain TCP, plan separate ports or addresses before configuring the senders.
5. After changing the input settings, update the senders so they use the new destination port or IP address.

### Related information

- What do CA PEM, Certificate PEM, and Key PEM mean for TLS input services?
- Use the product-specific service reference page for the exact UI fields of the input type you are configuring.

## What do CA PEM, Certificate PEM, and Key PEM mean for TLS input services?

### Question

What do the `CA PEM`, `Certificate PEM`, and `Key PEM` fields mean for TLS-enabled input services such as `RELP Listener` or `SETP Server`?

### Answer

For TLS-enabled input services:

- `CA PEM` is the CA bundle used to validate peer certificates.
- `Certificate PEM` is the certificate presented by the receiving service.
- `Key PEM` is the private key that matches that certificate.

The private key must be in PEM format and must not be protected by a passphrase.

### Details

The `CA PEM` field is used when certificate validation is enabled for incoming TLS connections. The receiving service uses the certificates in this file to validate certificates presented by connecting clients.

The `Certificate PEM` field should contain the certificate that the product presents to connecting clients. If intermediate CA certificates are required so that clients can build the chain, include the service certificate first and then append the intermediate CA certificates in order toward the root CA.

The `Key PEM` field must contain the private key that belongs to that certificate. Use an unencrypted PEM key file. Passphrase-protected private keys are not supported in these fields.

If client certificates are issued through a CA chain, the `CA PEM` file can contain multiple CA certificates. Include the intermediate CA certificates first, followed by the root CA certificate.

### Action path

1. Decide whether the input service should validate client certificates.
2. Prepare a PEM CA bundle for `CA PEM` if certificate validation is used.
3. Prepare a PEM certificate file for `Certificate PEM`. Include the intermediate chain if clients need it to validate the certificate.
4. Prepare the matching PEM private key for `Key PEM` and remove any passphrase protection before using it.
5. Load the files into the TLS configuration fields and test the connection.

### Related information

See the product-specific TLS service reference page for the exact UI location of these fields.

## Forwarding syslog messages with original IP address in WinSyslog

This article explains why forwarded syslog messages show the forwarding server’s IP address instead of the original device’s IP address and how to preserve the original source information in WinSyslog.

### Problem

When WinSyslog receives a syslog message and forwards it to another server, the forwarded message shows WinSyslog’s IP address as the source instead of the original device’s IP address. This makes it difficult to identify the true source of log messages in multi-hop forwarding scenarios.

### Root Cause

This is a limitation of the traditional UDP-based syslog protocol (RFC 3164). The IP header of a forwarded UDP packet always reflects the address of the forwarding server, not the original sender. The syslog protocol itself does not mandate that the original source address be preserved in the message body, so the information is lost unless additional steps are taken.

### Solutions

#### Option 1: Enable RFC 3164 parsing

If the originating devices are RFC 3164 compliant, WinSyslog can extract the hostname from the syslog message header rather than relying on the IP header.

1. Open the WinSyslog Configuration Client.
2. Navigate to the Syslog Server service.
3. Enable **RFC 3164 Parsing**.
4. Save the configuration and allow WinSyslog to reload.

> **Note**
>
> Many devices are not fully RFC 3164 compliant. Incorrectly formatted headers can result in wrong hostnames being parsed.

#### Option 2: Use the RELP protocol

RELP (Reliable Event Logging Protocol) is a TCP-based protocol that provides guaranteed delivery and preserves the original host information.

1. Open the WinSyslog Configuration Client.
2. Navigate to your rule or create a new one.
3. Add a new action and select **Forward Via RELP**.
4. Configure the following settings:
    - **Remote Host:** IP address or hostname of the RELP server.
    - **Port:** 20514 (default RELP port).
    - Enable **Preserve Original Hostname** if available.
    - Enable **TLS/SSL** if secure transport is required.
5. Test the connection to verify the RELP handshake.
6. Save the configuration and restart the WinSyslog service.

Benefits of RELP:

- Guaranteed message delivery with acknowledgments.
- Automatic retry for failed transmissions.
- TLS encryption for secure transport.
- Cross-platform compatibility with Unix/Linux receivers.

#### Option 3: Use the SETP protocol

SETP (Secure Event Transfer Protocol) is a proprietary Adiscon protocol designed for Windows-to-Windows log forwarding. It preserves all message fields, including the original source.

1. Configure a SETP listener on the receiving WinSyslog instance.
2. On the forwarding WinSyslog instance, add a **Forward via SETP** action.
3. Enter the IP address and port of the receiving server.
4. Save the configuration and restart the WinSyslog service.

> **Note**
>
> SETP requires both sender and receiver to be Adiscon products. It is best suited for internal Windows-to-Windows environments.

### Option 4: Enable Include Original Host

Add a tag at the beginning of the forwarded message that contains the original host information.

1. Open the WinSyslog Configuration Client.
2. Navigate to your syslog forwarding action.
3. Enable **Include Original Host**.
4. Save the configuration and allow WinSyslog to reload.

This adds a tag such as `FromHost: <ip>` at the start of the message.

> **Note**
>
> This approach is not RFC 3164 compliant. The receiving server must be able to parse the additional tag to extract the original host.

### Option 5: Forward in XML format

Forwarding messages in XML format preserves all original metadata, including the source IP.

1. Open the WinSyslog Configuration Client.
2. Navigate to your forwarding action.
3. Select **XML Format** as the output format.
4. Save the configuration and allow WinSyslog to reload.

> **Note**
>
> XML-formatted messages are not standard syslog. The receiving system must support XML parsing.

### Recommendations

| Scenario | Recommended Solution |
|---|---|
| Production environments requiring guaranteed delivery | RELP protocol |
| Internal Windows-to-Windows forwarding | SETP protocol |
| RFC 3164 compliant network devices | Enable RFC 3164 parsing |
| Quick compatibility fix for non-compliant devices | Include Original Host option |
| Automated parsing systems | XML format forwarding |
| Compliance and audit requirements | RELP with TLS encryption |

### Verification

1. Forward a test syslog message through WinSyslog.
2. On the receiving server, verify that the original source IP or hostname appears instead of WinSyslog's IP address.
3. Check the Windows Application Event Log on the WinSyslog server for any forwarding errors.
4. If using RELP, confirm that the connection handshake completes and messages are acknowledged.

## What Is the Difference Between SETP and Syslog?

### Question

What is the difference between SETP and syslog, and when should I use each one with WinSyslog?

### Answer

Use syslog when you need broad interoperability with network devices and standard log senders. Use SETP when both sides are Adiscon products and you need reliable transfer of richer event data.

### Details

**Syslog** is the standard protocol family for log transport. It is widely supported by routers, switches, firewalls, Linux systems, appliances, and many applications. It is the default choice whenever compatibility matters most.

**SETP** is an Adiscon-specific protocol for transferring events between compatible Adiscon products. It is designed to preserve more event properties than traditional syslog and is better suited for Windows-to-Windows Adiscon scenarios.

### Comparison

| Topic | Syslog | SETP |
|---|---|---|
| Compatibility | Broad industry support | Adiscon-product specific |
| Typical transport | UDP, TCP, TLS depending on sender and receiver | Adiscon protocol stack |
| Reliability | Depends on transport and sender implementation | Intended for reliable Adiscon-to-Adiscon transfer |
| Event detail preservation | Varies by sender format and parsing | Better preservation of original event properties |
| Best fit | Devices and mixed environments | Internal Adiscon product chains |

### Action path

Choose **syslog** when:

- logs come from network devices or non-Adiscon systems
- interoperability is the main requirement
- the receiver is not an Adiscon product

Choose **SETP** when:

- both sender and receiver are Adiscon products
- you want better preservation of Windows event details
- you control both ends of the connection

### Related information

- SETP Server
- Tutorial: Configure a SETP Server Service
- Receive Logs

## Why can WinSyslog use EventReporter properties?

### Question

Why can WinSyslog rules use EventReporter or Windows Event Log properties when WinSyslog does not collect Windows Event Logs locally?

### Answer

WinSyslog can use those properties when the event arrives through SETP. SETP preserves structured Adiscon event properties, so a WinSyslog ruleset can filter, format, and forward those properties after receiving the event.

### Details

This applies to events sent by another Adiscon product, such as EventReporter, to a WinSyslog `SETP Server` service. WinSyslog receives the structured event and processes it through the configured ruleset.

This does not mean that WinSyslog provides EventReporter's local Windows Event Log Monitor service. EventReporter collects the Windows Event Log event; WinSyslog can process the transported event properties after SETP delivery.

The `SETP Server` service is available in WinSyslog Enterprise and Trial. Sending events through SETP is a separate sender-side action.

### Related information

- What Is the Difference Between SETP and Syslog?
- Tutorial: Configure a SETP Server Service
- SETP Server

## How Do I Handle Non-Standard Syslog Messages from Unix or Linux Systems?

### Question

How should I handle syslog messages from Unix or Linux systems when the sender uses a non-standard format that WinSyslog does not parse correctly?

### Answer

If the sender does not produce RFC-compliant syslog, disable strict parsing for that stream and treat the payload more defensively. The cleanest fix is still on the sender side: make it emit standards-compliant syslog if possible.

### Details

Some Unix or Linux senders emit messages that do not fully match RFC 3164 or RFC 5424 expectations. A common example is a message that omits the hostname in positions where the parser expects it. In those cases, WinSyslog may interpret parts of the message incorrectly.

### Best-practice approach

1. Fix the sender format if you control the source system.
2. If that is not possible, reduce dependence on strict header parsing.
3. Apply parsing, filtering, or normalization later in the processing chain.

### Operational guidance

When you see incorrectly parsed hostnames, empty tags, or broken field extraction:

- inspect a raw sample message
- verify whether the sender actually follows RFC 3164 or RFC 5424
- adjust the WinSyslog service configuration so it does not rely on a header format the sender does not provide
- use downstream filtering or custom property extraction where necessary

### Related information

- Syslog server
- Receive Logs
- Syslog Message Properties

# Licensing and product positioning

## How Do I Enter WinSyslog License Information?

### Answer

Apply the `license.alic` file through **General** -> **License** in the configuration client, save, and restart the service.

### Details

Current WinSyslog builds require a `license.alic` file from Adiscon.

Default file location:

`%ProgramData%\Adiscon\WinSyslog\license.alic`

You can set `szLicenseV2Path` to another path if you do not use the default location.

### Action path

1. Obtain `license.alic` from Adiscon for your edition.
2. Open the WinSyslog Configuration Client.
3. Expand **General** and select **License**.
4. Browse for `license.alic`, drag-and-drop the file, or paste the file path.
5. Save the configuration and verify license status in the client.
6. Restart the WinSyslog service.

### Verification

After the service restart, reopen the license page and confirm that license status is shown without validation errors.

### Related information

- What is Freeware Mode?
- Installation
- How do I contact Adiscon sales?

## What is Freeware Mode?

### Overview

This FAQ explains WinSyslog's Freeware Mode, which provides free functionality for home users while offering advanced capabilities in the Professional edition.

### What is Freeware Mode?

**Adiscon cares for the home user.** There are two versions of WinSyslog - the "InterActive SyslogViewer" and "WinSyslog Professional". The free edition is targeted to the home user. It offers a free, interactive scrolling display of received syslog messages in InterActive SyslogViewer. The 60 most current messages can be viewed in this mode. There is no time limit for this functionality. This mode is referred to as "Freeware Mode".

**The Professional edition offers advanced capabilities** like logging to file or database and an enhanced rule processing engine.

### How Does Licensing Work?

**After installation, the product starts as Professional version.** If the professional version is not purchased within 30 days, the product automatically switches to freeware mode. So there is no need to worry about unlicensed usage in this case.

### Key Features of Freeware Mode

- **Interactive message display**: View the 60 most current syslog messages
- **No time limit**: Use the freeware mode indefinitely
- **Free support**: Even free copies come with Adiscon's great support
- **Home user focused**: Perfect for personal and home environments

### Professional Edition Benefits

For users requiring advanced capabilities, the Professional edition provides:

- **File logging**: Persistent storage of syslog messages to files
- **Database logging**: Store events in databases via ODBC
- **Enhanced rule processing**: Powerful filtering and event processing engine
- **Advanced actions**: Email notifications, forwarding, and more

### Automatic Mode Transition

The transition from Professional to Freeware Mode is automatic and hassle-free:

1. WinSyslog starts in Professional mode after installation
2. A 30-day evaluation period is provided
3. If not licensed within 30 days, it automatically switches to Freeware Mode
4. No manual intervention required - the system handles the transition

### Notes

For more information about WinSyslog features and capabilities, see the `../features` documentation.

## WinSyslog vs Kiwi Syslog Server – Which to Choose?

### Overview

Many Windows administrators evaluate syslog server alternatives for their network logging needs. This FAQ helps you compare Kiwi Syslog Server with WinSyslog based on verified information from official documentation.

### What Administrators Are Looking For

Organizations typically seek syslog solutions that provide:

- Reliable message parsing and processing
- Straightforward configuration management
- Responsive technical support
- Competitive licensing options
- Proven track record and ongoing development

### Historical Context

**WinSyslog** was the first syslog server for Windows, originally developed in 1996. It has been actively maintained by the same engineering team for over 25 years. WinSyslog is developed by Adiscon, the same company behind rsyslog, ensuring deep protocol understanding and consistent implementation across platforms.

**Kiwi Syslog Server** has been available since the early 2000s, offering a GUI-based approach to syslog management.

### Feature Comparison

| Feature | Kiwi Syslog Server | WinSyslog |
|---|---|---|
| Windows Syslog History | Available since ~2000s | First syslog server for Windows (1996) |

| | | |
|---|---|---|
| Multiple Instances | Not supported | Full support for multiple independent servers |
| Message Parsing | RFC 3164 compliant | RFC 3164 and RFC 5424 compliant |
| Configuration | Filter-based | Rule-based processing |
| Rule Processing | Filters + actions | Flexible rules + actions architecture |
| Database Support | ODBC databases (e.g., SQL Server) | Any ODBC-compliant database |
| RFC 5424 Support | Not supported | Full compliance |
| RELP Support | Not supported | Yes – Reliable Event Logging Protocol |
| TLS Support | Yes (for secure TCP) | Yes – Built-in secure transport |
| rsyslog Integration | Basic forwarding | Seamless integration |
| Windows Event Log | Built-in action | Advanced built-in integration |
| Design Focus | GUI-based management | Reliability and automation |

### Protocol Support Comparison

| Protocol | Kiwi Syslog Server | WinSyslog |
|---|---|---|
| UDP | Yes | Yes |
| TCP | Yes | Yes |
| TLS | Yes (for secure TCP) | Yes – Built-in |
| RELP | Not supported | Yes |
| Message Parsing | RFC 3164 | RFC 3164, RFC 5424 |

### Licensing Comparison

| Aspect | Kiwi Syslog Server | WinSyslog |
|---|---|---|
| Free Version | Freeware available | Trial version available |
| Licensed Version | Subscription-based | One-time or competitive licensing |
| UpgradeInsurance | Not applicable | Available option |
| Trial | Limited functionality | Fully functional |
| Support | Various options | Support Portal (ticket.adiscon.com) |

### Support and Development

| Aspect | Kiwi Syslog Server | WinSyslog |
|---|---|---|
| Support Channel | Various | Support Portal (ticket.adiscon.com) |
| Response Time | Varies | 1-2 business days, often same day |
| Development Model | Vendor-driven | User-driven (quarterly feature implementation) |
| Product Updates | Regular releases | Regular releases with UpgradeInsurance option |

### Key WinSyslog Advantages

WinSyslog offers several key advantages:

- **Built by the original team behind rsyslog** – Provides deep syslog protocol expertise
- **Designed for professionals who care about results** – Focus on reliability over flashy interfaces
- **25+ years of proven stability** – Established track record in enterprise environments

- **Multiple syslog server instances** – Unique capability to run multiple independent servers on one machine
- **Standards-compliant parsing** – Full RFC 3164 and RFC 5424 support
- **Rule-based configuration** – More flexible than filter-based approaches
- **Comprehensive database integration** – Support for any ODBC-compliant database
- **RELP and TLS support** – Secure transport options built-in
- **Seamless rsyslog integration** – Works seamlessly with rsyslog backends

### Migration Benefits

Organizations migrating to WinSyslog benefit from:

- Rule-based configuration approach
- Standards-compliant message parsing
- Proven performance track record
- Flexible automation capabilities

### Trial and Evaluation

WinSyslog offers a fully functional trial version with no obligation. You can:

- Download and test in your environment
- Evaluate against your current pain points
- Contact support via the Support Portal
- Get responses within 1-2 business days (often same day during German office hours)

### Reliability and Track Record

WinSyslog has been:

- The first syslog server for Windows (since 1996)
- Maintained by the same engineering team throughout its history
- Used by enterprises worldwide
- Proven at customer sites for decades

### Decision Checklist

Consider WinSyslog if you need:

- Reliable message parsing
- Straightforward configuration
- Responsive technical support
- Cost-effective solution
- Predictable, reliable syslog server
- Professional-grade automation capabilities

### Next Steps

Ready to evaluate WinSyslog?

1. Download trial version
2. Contact us via the Support Portal
3. Test in your environment
4. Evaluate against your current requirements

## Why did product versions jump to 26.07?

### Question

Why is the current version `26.07` when older product versions used numbers such as 18, 19, 15, or 8?

### Answer

Adiscon changed to calendar-based version numbers. In `26.07`, `26` means 2026 and `07` means July. There are no missing intermediate product releases.

### Details

The latest old-style version families were:

- **WinSyslog**: `18.x`
- **EventReporter**: `19.x`
- **WinSyslog**: `15.x`
- **rsyslog Windows Agent**: `8.x`

The new `YY.MM` format starts with `26.07`.

### Related information

- See the product website for version history and release notes.

## What happens when the client opens a configuration with unsupported blocks?

### Question

The configuration client shows a warning about unsupported configuration items when I open a file. What does that mean?

### Answer

The client skipped configuration blocks that this client version does not understand. Saving the configuration may remove those unsupported sections.

### Details

Newer service or file formats can introduce configuration blocks that a configuration client cannot edit. The client loads what it supports and displays a one-time warning.

Current configuration clients preserve an existing `general(name="Metrics")` block when they load and save CFG, YAML, or registry configuration. They do not provide a Metrics editor. Other unknown blocks can still be skipped and may be lost when the configuration is saved.

If you save without upgrading the client, unsupported blocks may be dropped from the saved file.

### Action path

1. Note which blocks the warning mentions.
2. If you need to edit those settings, upgrade to a configuration client that supports them, or edit the YAML or legacy `.cfg` file with care in a text editor.
3. Back up the configuration before saving from an older client.

### Related information

- See the file-based configuration chapter in the WinSyslog, EventReporter, WinSyslog, or rsyslog Windows Agent manual.

## Why is Net Send no longer available in the configuration client?

### Question

I cannot add a Net Send action in the configuration client. Where did it go?

### Answer

Net Send was **removed from the configuration client** in the **26.07** release for WinSyslog, EventReporter, and WinSyslog.

### Details

The Windows Messenger service (`net send`) is deprecated on modern Windows and was unreliable for production alerting. The **26.07** client no longer offers Net Send in the action list.

Existing configurations that still reference Net Send may load with a skip or warning depending on client version. You cannot create new Net Send actions from the current client.

### Action path

Use one of these alternatives in your ruleset:

- **Send Email** for operator notifications.
- **HTTP REST Output** for webhooks and REST endpoints (see **Actions** → **Forwarding** in this manual).
- **Start Program** to invoke a custom script or notifier.
- Syslog or RELP forwarding to a central monitoring system.

### Related information

For the deprecated Net Send action reference and HTTP REST Output details, see **Configuration** → **Actions** in the WinSyslog, EventReporter, or WinSyslog manual.

## How do I set up HTTP REST Output?

### Question

How do I forward events to a REST API or OpenSearch using the configuration client?

### Answer

Add an **HTTP REST Output** action to your ruleset, choose a profile (Custom or OpenSearch bulk), configure the endpoint and request body, and save the configuration.

### Details

The **HTTP REST Output** action (type **1032**, `RestOutput`) is available on WinSyslog, EventReporter, and WinSyslog starting with **26.07**. It is listed under **Forwarding** in the action list.

### Action path

1. Open the configuration client and select the target ruleset and rule.
2. Add action → **HTTP REST Output**.
3. On the **General** tab, choose **Custom** or **OpenSearch bulk** and set the endpoint (server, path, port, TLS as needed).
4. On the **Request** tab, set the HTTP method, content type, headers, and request template or JSON property mappings.
5. On the **Advanced** tab, adjust timeout, user agent, and retry status codes if needed.
6. Save the configuration and restart the service.
7. Send a test event and verify delivery at the remote endpoint.

### Related information

- See **HTTP REST Output** and **HTTP Request** under **Actions** → **Forwarding** in the WinSyslog, EventReporter, or WinSyslog manual when those actions are part of your product build.

- Accessing Properties

## Platform and compatibility

### Do the configuration clients require .NET Framework, or is .NET Core or .NET 5+ enough?

#### Question

Can the Windows configuration clients run with only the newer .NET runtime installed, such as .NET Core, .NET 8, or .NET 10?

#### Answer

No. The Windows configuration clients require **Microsoft .NET Framework 4.7.2 or a newer .NET Framework 4.x release**.

`.NET Framework 4.x` is a Windows-only runtime family. `.NET Core` and `.NET 5+` are a different cross-platform runtime family.

`.NET 5+` continues the `.NET Core` line under a new name. It is not a newer version of `.NET Framework 4.x`.

Installing only `.NET Core` or `.NET 5+` does not satisfy a `.NET Framework` requirement.

#### Details

For these products, the requirement applies to the **Windows configuration client**. The background service is a separate component.

If you deploy only the background service on a target system, the Configuration Client is not required on that system. In that case, this `.NET Framework` requirement applies where the Configuration Client is installed and used, not to the service-only target.

The following satisfy this requirement:

- `.NET Framework 4.7.2`
- `.NET Framework 4.8`
- `.NET Framework 4.8.1`

The following do not satisfy this requirement on their own:

- `.NET Core 3.x`
- `.NET 5`
- `.NET 6`
- `.NET 8`
- `.NET 10`

#### Action path

1. Check whether the system has `.NET Framework 4.7.2` or a newer `.NET Framework 4.x` release installed.
2. If it is missing, install `.NET Framework` separately or allow the product installer to add the required Framework components.
3. If the system only has `.NET Core` or `.NET 5+`, do not assume that the configuration client can run.

#### Related information

- Microsoft documentation on .NET Framework versions and dependencies: https://learn.microsoft.com/en-us/dotnet/framework/install/versions-and-dependencies
- Microsoft documentation on installing .NET on Windows: https://learn.microsoft.com/en-us/dotnet/core/install/windows

## Is WinSyslog v18+ supported on Windows Server IoT 2025?

### Overview

This FAQ answers whether WinSyslog v18+ is supported on Windows Server IoT 2025 and outlines current guidance, functional status, and considerations for deployments, including Server Core.

### Support Status

**Official support:**

- Windows Server IoT 2025 is not yet explicitly listed in the WinSyslog v18+ platform matrix.

**Functional status:**

- WinSyslog v18+ is known to function properly on Windows Server IoT 2025 (including Server Core) based on internal testing and field feedback.

### Guidance for Server Core Deployments

Windows Server IoT 2025 Server Core does not provide a graphical user interface. For new file-based deployments, use YAML configuration (`.yaml`) and point `szFileConfig` to that file. The legacy `.cfg` Adiscon Config Format remains supported for compatibility with existing installations, test fixtures, support workflows, and external tooling, but it is not required for new deployments.

Recommended workflow:

1. Create the configuration on a GUI-enabled machine
   - Install WinSyslog and open the Configuration Client
   - Configure rules, services, and actions as required
   - Export YAML (`.yaml`) for Server Core file config mode, backup, or re-import on another GUI machine
2. Transfer the configuration to Server Core
   - Copy the exported `.yaml` file to the Server Core system (e.g., via PowerShell Remoting or SMB)
3. Enable File Config Mode and set paths via registry

   Registry path: `HKEY_LOCAL_MACHINE\SOFTWARE\Adiscon\WinSyslog\Settings`

   Required values:

   - `szFileConfig` (REG_SZ): Example `c:\configs\winsyslog\central-server.yaml`
   - `szDataDirectory` (REG_SZ): Example `c:\configs\winsyslog\`
   - `iAccessMode` (REG_DWORD): `1` (enables file config mode)

> **Important**
> When running in file config mode, ensure the service account has read access to the configuration file and write access to the data directory.

### Notes

# Sales

This section answers sales and licensing questions for Adiscon products in a question-focused format.

Use this section for commercial topics (quotes, orders, licenses, renewals, and purchase orders). For technical troubleshooting and product support, use the FAQ and troubleshooting sections in the relevant product manual.

## Quick actions

- I need to contact sales: How do I contact Adiscon sales?.

- I need to request a quote: What should I include in a quote request?.
- I opened a ticket and need to check delivery/notification behavior: What happens after I open a sales ticket?.
- I need help with purchase orders and billing data: How do purchase orders and billing requests work?.
- I need product licensing and ordering links: Licensing and ordering.
- I need to use an older product version with my license: Can I use an older product version with my license?.
- I need to confirm offline or air-gapped licensing behavior: Air-gapped environments.
- I need UpgradeInsurance policy details: UpgradeInsurance.

## How do I contact Adiscon sales?

### Short answer

Use the Customer Service System as the primary channel; email to `sales@adiscon.com` is also accepted and creates tickets.

### Question

How do I contact Adiscon sales for quotes, ordering, licensing, or renewals?

### Answer

Use the Customer Service System. This is the primary contact path for sales questions.

### Details

- You can also send email to `sales@adiscon.com`.
- Emails to `sales@adiscon.com` create tickets automatically.
- The ticket portal is recommended because email delivery can be unreliable in both directions (customer to Adiscon and Adiscon to customer).
- If you do not see a response, this is usually a delivery or filtering issue, not an intentionally unanswered request.
- Choosing a sales category helps speed up handling. If a ticket is opened in the wrong category, it is reassigned internally.
- For language handling, see **`../contact-language-policy`**.

### Technical support routing

For technical incidents, troubleshooting, and product behavior issues, use the support category in the same system or the FAQ and troubleshooting sections in your product manual.

### Next action

Open a sales ticket in the Customer Service System or send email to `sales@adiscon.com` (emails create tickets).

## What should I include in a quote request?

### Short answer

You can start with a short request, and sales will guide you through missing details; providing more context upfront can speed up the first quote.

### Question

What information should I provide to get an accurate sales quote quickly?

### Answer

You can start with a short message like “I want to buy your tool.” Sales will follow up and guide you through any missing details.

### Details

#### If you want a faster first quote

- Product name
- Edition (if known) or note that you need help selecting one
- Number of licenses or systems/devices to be covered
- Billing country
- Commercial contact email

#### Optional but speeds finalization

- Deployment type (for example production, test, or mixed)
- License term and UpgradeInsurance requirements
- Preferred purchase method (online order or purchase order)
- Company legal name and billing contact details
- Target purchase timeline
- Existing license, renewal, or expansion context

#### EU VAT note

Customers in EU member states are normally charged VAT. If your organization is eligible for VAT exemption, provide your official VAT ID during ordering or invoicing so the billing team can apply the exemption.

#### Optional copy/paste template

If you already have details, this template can speed up quote processing:

```
Product:
Edition (or "help me choose"):
Scope (licenses/systems/devices):
Deployment type (production/test/mixed):
License term / UpgradeInsurance needs:
Billing country:
Company legal name:
Purchase method (online order/PO):
Target purchase date:
Existing license context (if any):
VAT ID (EU, if applicable):
Contact email:
```

#### Next action

Open a sales ticket in the Customer Service System or send email to `sales@adiscon.com` (emails create tickets).

## What happens after I open a sales ticket?

#### Short answer

Your ticket is confirmed by email and then handled in the same ticket thread, with first response usually within one business day.

#### Question

What should I expect after opening a sales ticket?

#### Answer

Adiscon sales reviews and responds in the same ticket thread, which is the primary place to track status and replies.

### Normal flow

1. You open a ticket (via portal or email).
2. An automatic confirmation email is sent (if a valid address is available).
3. A sales agent replies in the same ticket thread.
4. Follow-up questions and updates continue in that same thread.

### Details

#### Notification behavior

When a ticket is opened, an automatic response is sent by email to the registered address (if available).

#### If no automatic response arrives

Do the following:

- Spam or junk folders
- Mail security gateways
- Internal email policy filters

Then sign in to the Customer Service System to confirm whether your ticket was created.

#### Avoid duplicate tickets

If a ticket already exists, continue in that ticket instead of opening a duplicate request. Duplicate tickets can slow handling and split context.

When replying by email, keep the same subject/thread whenever possible so updates remain attached to the original ticket.

### Next action

Check and continue the existing ticket in the Customer Service System. If no ticket exists, open a new one in the portal or send email to `sales@adiscon.com`.

## How do purchase orders and billing requests work?

### Short answer

Use the Customer Service System for PO processing, billing changes, and invoice requests; for enterprise procurement flows, describe your process and Adiscon will adapt as much as possible.

### Question

How do I handle purchase orders, invoicing, and billing-data updates?

### Answer

Use the Customer Service System for purchase order processing and billing requests. Include complete company and billing details to avoid order delays.

### Details

#### Typical flow

1. Open a sales ticket in the Customer Service System.
2. Share PO and billing details in that ticket.
3. Sales confirms details and clarifies open points.
4. Order and invoice handling continues in the same ticket thread.

For larger organizations with their own procurement flow, open a sales ticket and describe your process. Adiscon will adapt as much as possible.

### Purchase orders

Helpful information to include at the start:

- PO number (if already issued)
- Company legal entity
- Billing address
- Billing contact
- Product and scope

For formal purchase order requirements and additional details, see: Adiscon purchase orders.

### Billing and invoice requests

For invoice details, billing data updates, tax/VAT information, and related order handling, contact sales via the Customer Service System.

Use the Customer Service System for billing-data changes both before and after order placement.

### Security and language

- Prefer the ticket portal (including attachments) over plain email when sending sensitive billing data.
- For language handling, see **`../contact-language-policy`**.

### Common requests

The sales team can also handle:

- Invoice copy and invoice re-send requests
- Billing address/contact changes
- VAT/tax-related billing updates

### Next action

Open a sales ticket in the Customer Service System or send email to `sales@adiscon.com` (emails create tickets).

## Licensing and ordering

### Short answer

Use product ordering pages for direct purchase or open a sales ticket for edition guidance and licensing clarification.

### Question

How do I order Adiscon products, and where do I clarify licensing questions?

### Answer

Order through the product-specific ordering pages or via purchase order. If edition or licensing details are unclear, open a sales ticket first.

### Details

#### Trial period

WinSyslog, EventReporter, WinSyslog, and rsyslog Windows Agent provide 30-day trial functionality after installation. After that period, a valid license is required for full-feature operation.

### License agreement (EULA)

The End User License Agreement (EULA) is shown during setup. If you need a copy or have licensing questions, contact the Customer Service System.

### Product-specific links

| Product | Pricing and ordering |
|---|---|
| WinSyslog | WinSyslog ordering page |
| EventReporter | EventReporter ordering page |
| WinSyslog | MonitorWare ordering page |
| rsyslog Windows Agent | rsyslog Windows Agent ordering page |

### Choosing an edition

If you are unsure which edition fits your requirements, open a sales ticket in the Customer Service System. A short request is enough to start.

### Product-specific licensing details

Licensing and counting rules can be product-specific. For now, use the sales ticket process to clarify product-specific licensing details. These details will be documented in dedicated product-specific sales pages.

### Offline and licensing behavior

For common licensing and deployment questions, see these canonical answers:

- Air-gapped environments
- Offline installation and activation
- Online verification after activation
- Perpetual licenses and UpgradeInsurance
- Can I use an older product version with my license?

### Purchase methods

You can place orders through each product ordering page or by purchase order. If you need a formal quote before placing the order, open a sales ticket in the Customer Service System.

### Maintenance and renewal

For maintenance and renewal policy details, see UpgradeInsurance.

### Purchase orders and billing

For purchase order handling and billing details, see How do purchase orders and billing requests work?.

### Next action

Open a sales ticket in the Customer Service System or send email to `sales@adiscon.com` (emails create tickets).

## Can I use an older product version with my license?

### Question

Can I use an older Adiscon product version with my current license?

### Answer

Yes. Your license can cover older product versions.

If the older product version does not support license files, contact Adiscon support or sales and request a free license key for that product version.

### Details

Current products use a `license.alic` license file. Older products that do not support license files use a registration name and license key instead. Tell Adiscon which product and version you need to run so the license can be issued in the right format.

### Related information

- How do I contact Adiscon sales?
- Licensing and ordering

## Air-gapped environments

### Short answer

Adiscon products can be deployed and operated in fully closed environments without internet access.

### Question

Can Adiscon products be used in completely closed or air-gapped environments?

### Answer

Yes. Adiscon products support deployment and operation in fully closed networks.

### Details

Products can be installed and operated in environments where internet access is not available or not permitted. They do not require outbound internet access for normal runtime operation in such environments.

If you also need details about license activation without internet access, see Offline installation and activation.

### Action path

If your procurement or security process requires a written confirmation for a specific environment, open a sales ticket in the Customer Service System.

### Related information

- Offline installation and activation
- Online verification after activation

## Offline installation and activation

### Short answer

Installation and license activation do not require internet access.

### Question

Do Adiscon product installation and license activation require internet access?

### Answer

No. Installation and license activation can be completed without an internet connection.

### Details

Adiscon supports installation and license activation in offline environments. This allows products to be deployed in restricted networks where direct internet connectivity is not available.

This question is separate from whether the licensed product later requires online verification. For that, see Online verification after activation.

### Action path

If you need product delivery or license handling guidance for an offline rollout, open a sales ticket in the Customer Service System.

### Related information

- Air-gapped environments
- Online verification after activation

## Online verification after activation

### Short answer

After activation, Adiscon product licenses do not require ongoing internet connectivity, online heartbeats, or “phone home” checks to remain valid.

### Question

Do Adiscon product licenses require online verification after activation?

### Answer

No. After activation, the license remains valid without periodic online verification.

### Details

Once the product is installed and the license has been applied, no ongoing outbound connection to Adiscon systems is required to keep the license active. There is no periodic online verification, heartbeat, or “phone home” mechanism needed to maintain licensed status.

This question is separate from whether installation and activation themselves require internet access. For that, see Offline installation and activation.

### Action path

If your security or compliance process requires explicit confirmation about offline license behavior, open a sales ticket in the Customer Service System.

### Related information

- Air-gapped environments
- Offline installation and activation
- Perpetual licenses and UpgradeInsurance

## Perpetual licenses and UpgradeInsurance

### Short answer

Adiscon product licenses are perpetual. UpgradeInsurance is a separate, time-limited service that is strongly recommended for access to version-line upgrades.

### Question

Are Adiscon product licenses perpetual, and how does that differ from UpgradeInsurance?

### Answer

Adiscon product licenses are perpetual. UpgradeInsurance is an additional, time-limited service for upgrade and maintenance benefits.

### Details

A perpetual license allows continued use of the licensed product version after activation. It does not depend on periodic online verification to remain valid.

UpgradeInsurance is separate from the base license. It is a time-limited additional service that is strongly recommended if you want access to future product version lines and related maintenance benefits.

When UpgradeInsurance expires, the licensed product version does not stop working. What changes is entitlement to later version-line upgrades and the additional benefits covered by active UpgradeInsurance.

**License files:** Current product versions use a `license.alic` file from Adiscon.

For detailed renewal and reinstatement policy, see UpgradeInsurance.

### Action path

If you need confirmation for purchasing, renewal, or compliance review, open a sales ticket in the Customer Service System.

### Related information

- UpgradeInsurance
- Can I use an older product version with my license?
- Online verification after activation

## UpgradeInsurance

### Short answer

UpgradeInsurance provides version-line upgrade rights and priority support while active; if coverage has lapsed, reinstatement is possible with back-dated start rules.

### Question

What is UpgradeInsurance, and how do renewal and reinstatement work?

### Answer

UpgradeInsurance is Adiscon’s maintenance plan.

### Details

### What it includes

- Free version-line upgrades during active coverage

- Priority support handling

### Coverage period

UpgradeInsurance is available for periods between 1 and 5 years.

### Coverage continuity and reinstatement

- UpgradeInsurance should be kept active through timely renewal.
- If coverage lapses, reinstatement is possible.
- If UpgradeInsurance was not purchased with the original license, it can still be added later.
- Reinstatement is back-dated to the previous coverage end date or, if it was never purchased, to the original license purchase date.

### Difference from the base license

UpgradeInsurance is separate from the base product license. For the distinction between perpetual licenses and this time-limited service, see Perpetual licenses and UpgradeInsurance.

### Examples

1. If coverage ended on June 30 and renewal is requested in September, reinstatement starts from June 30.
2. If no UpgradeInsurance was purchased with the initial order, later activation is back-dated to the original license purchase date.

### How to request renewal or reinstatement

Open a sales ticket in the Customer Service System or send email to `sales@adiscon.com` (emails create tickets).

### Next action

Open a sales ticket in the Customer Service System or send email to `sales@adiscon.com` (emails create tickets).

# Reference

Use this section for lookup material, not for first-time setup. It collects the pages you typically need while operating, troubleshooting, or fine-tuning an existing WinSyslog deployment.

Use `Getting Started` for first-time setup, `Tutorials` for task-by-task guidance, and `Configuration` for the main product setup pages.

## Product controls and local operation

These pages cover product-specific controls and local administration details.

### WinSyslog Shortcut Keys

Use shortcut keys as an alternative to the mouse when working in WinSyslog. Keyboard shortcuts may also make it easier for you to interact with WinSyslog. All these shortkeys are usually available in textboxes only. Listed below are the available short keys:

CTRL+S = Save

CTRL+X = Cut

CTRL+C = Copy

CTRL+V = Paste

CTRL+Z = Undo

Note: This is in synchronization with most major Windows applications.

## Command Line Switches

There are several command line switches available for using WinSyslog from the command line. To use these switches you need administrative rights.

- `-h` Show command line help
- `-v` Show version information and whether the service is installed
- `-i` Install service
- `-u` Remove (uninstall) service
- `-i <CustomServiceName>` Install service with a custom service name
- `-u <CustomServiceName>` Uninstall a service with a custom service name
- `-r` Run as console application
- `-r -o` Run once as console application

If you install the service, you can start and stop it with commands such as `net start`, `net stop`, `sc start`, `sc stop`, or PowerShell (`Start-Service` / `Stop-Service`). By using the `-r` switch, you run it only on the command line. When you close the command line, the program will stop working.

The `-v` switch gives you information about the version of the service.

### Direct foreground configuration

This capability is available starting with **26.08**.

Foreground runs can use an ephemeral configuration without installing a service or creating a service `Parameters` key:

- `-r -fileconfig <path> [-datadir <path>]`
- `-r -accessmode file -fileconfig <path> [-datadir <path>]`
- `-r -accessmode registry [-regpath <path>]`

Direct options take precedence for that foreground process only. File mode requires `-fileconfig`; `-datadir` cannot be used alone or with explicit registry mode, and `-regpath` is available only with explicit registry mode. The existing `-i` installation configuration remains persistent and unchanged.

The first direct foreground run registers the canonical Windows Event Log provider using the executable message file. This does not create a service. Provider creation or an update can require one elevated first run; the command reports that requirement before startup.

**Custom service name examples:**

- `WINSyslog.exe -i CustomServiceName`
- `WINSyslog.exe -u CustomServiceName`

You can import configuration files via the command line as well. Use YAML (`.yaml`) for new imports. The legacy Adiscon Config Format (`.cfg`) remains supported for compatibility with existing files and support workflows. The syntax is quite easy: execute the WinSyslog configuration client and append the name of the configuration file.

**Sample:**

WINSyslogClient.exe example.yaml

or

WINSyslogClient.exe “example.yaml”

After this is executed, you will see the splash screen of the configuration client and then the import dialogue, which you have to confirm manually.

For doing a silent import, the `/f` parameter has to be appended. This will look like this:

```
WINSyslogClient.exe "example.yaml" /f
```

## Edition Comparison

WinSyslog comes in different versions. Some of them are more feature-richer than others. The manual covers description about the full feature set. In order to remove confusion we have created a Product Comparison Sheet which identifies the differences between different available versions. Click here to see which services, actions and other features are provided by each version.

**License file:** Current builds use a `license.alic` file.

# Technical lookup material

These pages provide low-level reference data that supports configuration and troubleshooting.

## Comparison of properties

**Available in WinSyslog, EventReporter and WinSyslog**

The property replacer is a reference. The actual properties available for an event depend on the product, the configured service, and in some cases the edition. For WinSyslog, Windows Event Log properties are available when the event is received through SETP from another Adiscon product.

| Properties Available | WinSyslog | WinSyslog | EventReporter |
|---|---|---|---|
| Standard Property | Yes | Yes | Yes |
| Windows Event Log | Yes | SETP | Yes |
| Syslog Message | Yes | Yes | |
| Disk Space Monitor | Yes | | |
| File Monitor | Yes | | |
| Windows Service Monitor | Yes | | Yes |
| Ping Probe | Yes | | |
| Port Probe | Yes | | |
| Database Monitor | Yes | | |
| Serial Port Monitor | Yes | | |
| MonitorWare Echo Request | Yes | | |
| System | Yes | Yes | Yes |
| Custom | Yes | Yes | Yes |
| NNTP Probe | Yes | | |
| HTTP Probe | Yes | | |
| FTP Probe | Yes | | |
| SMTP Probe | Yes | | |
| POP3 Probe | Yes | | |

`SETP` means that WinSyslog can use those properties when they are transported through SETP; it does not mean that WinSyslog collects Windows Event Logs locally.

## Event Properties

Events have certain properties, for example the message associated with the event or the time it was generated. Each of this properties has an assigned name. The actual properties available depend on the type of event. The following sections describe both how to access properties as well as properties available.

Knowing about event properties is important for building complex filter conditions, customized actions as well as for integrating into a third-party system. Event properties provide a generic way to look at and process the events generated. Thus we highly recommend that you at least briefly read this reference section.

### Accessing Properties

Properties are accessed by their name. The component used for this is called the “property replacer”. It is a generic component that allows you to merge properties from the event processed to e.g. the email subject line or a log file line. It is a central component that is used as often in the product as possible. The idea behind the property replacer is that there is often need to specify a value from the event processed.

The property replacer provides very powerful ways to access the properties: they cannot only be accessed as one full property. They can also be accessed as substrings and even be reformatted. As such, the property replacer provides a specific syntax to access properties:

`%property:fromPos:toPos:options%`

The percent-signs (“%”) indicate the start of a special sequence. The other parameters have the following meanings

FromPos and ToPos can be used to copy a substring from a lengthy property. The options allow to specify some additional formatting.

Within the properties, all time is based on UTC regardless if your preferred time is UTC or localtime. So if you want to display localtime instead of UTC, you have to use the following syntax: `%variable:::localtime%`

### Property

This is the name of the property to be replaced. It can be any property that a given event possesses. If a property is selected that is empty for the event processed, an empty string is returned.

A property is either an event property, a custom property, a dynamic property or a system property.

If a property is selected that is not present, the result will always be an empty string, no matter which other options have been selected.

### FromPos

If you do not want to use the full string from the property, you can specify a start position here. There are two ways to specify the start location:

**Fixed Character position**

If you know exactly on which position the string of interest begins, you can use a fixed location. In this case, simply specify the character position containing the first character of interest. Character positions are counted at 1.

**Search Pattern**

A search pattern is specified as follows:

/<search-pattern>/<options>

If a search pattern is specified, the property value is examined and the first occurrence of <search-pattern> is detected. If it is not found, nothing is returned. If it is found, the position where the pattern is found is the start position or, if the option “$” is specified, the position immediately after the pattern.

The search pattern may contain the “?” wildcard character, which represents any character. Other wildcards are not supported with the property replacer.

Please note that a slash inside the search pattern will terminate the search field. So pure slashes cannot be used. However, they can be escaped by prefixing them with a backslash (). The same applies to the ‘?’ character. For example, if you intend to search for `"http://"` inside a search pattern, you must use the following search string: `"/http:////"`.

**Default Value**

If the FromPos is not specified, the property string is copied starting at position 1.

### ToPos

If you do not want to use the full string from the property, you can specify the highest character position to be copied here.

**Absolute Position**

Specify a simple integer if you would like to specify an absolute ending position.

**Relative Position**

This is most useful together with the search capabilities of FromPos. A relative position allows you to specify how many characters before or after the FromPos you would like to have copied. Relative positions are specified by putting a plus or minus (“+”/”-”) in front of the integer.

Please note: if you specify a negative position (e.g. -20), FromPos and ToPos will internally be swapped. That is the property value will not be (somehow) reversely copied but they will be in right order. For example, if you specify `%msg:30:-20%` actually character positions 10 to 30 will be copied.

**Search Pattern**

Search pattern support is similar to search pattern support in FromPos.

A search pattern is specified as follows:

/<search-pattern>/<options>

If a search pattern is specified, the property value is examined and the first occurrence of <search-pattern> is detected. The search is only carried out in the string that follows FromPos. If the string is not found, nothing is returned. If it is found, the position where the pattern is found is the ending position or, if the option “$” is specified, the position immediately after the pattern.

The search pattern may contain the “?” wildcard character, which represents any character. Other wildcards are not supported with the property replacer.

Please note that a slash inside the search pattern will terminate the search field. So pure slashes cannot be used. However, they can be escaped by prefixing them with a backslash (). The same applies to the ‘?’ character. For example, if you intend to search for `"http://"` inside a search pattern, you must use the following search string: `"/http:////"`.

**Search Example**

A common use case is to combine searches in ToPos and FromPos to extract a substring that is delimited by two other strings. To do so, use search patterns in both fields. An example is as follows: assume a device might generate message in the form `"... error XXX occurred..."` where “…” represents additional message text and XXX the actual error cause. You would like to extract the phrase `"error XXX occurred"`. To do so, use the following property replacer syntax: `%msg:/error/:/occurred/$/%`

Please note that the FromPos is used without the $-option, while in ToPos it is used. If it hadn’t been used in ToPos, only the part “error XXX “ would have been extracted, as the ToPos would point to the last character before the search string.

Similarly, if only “ XXX “ should be extracted, the following syntax might be used:

```
%msg:/error/$:/occurred/%
```

If you would also like to remove the spaces (resulting in just “XXX”), you must include them into the search strings:

```
%msg:/error /:/ occurred/$/%
```

**Default**

If not specified, the ending position will be the last character.

### Options

Options allow you to modify the contents of the property. Multiple options can be set. They are comma-separated. If conflicting options are specified, always the last option will be in effect (e.g. specifying “uppercase,lowercase” will lead to lowercase conversion of the property value).

The following options are available with this release of the product:

**lowercase**

All characters in the resulting property extract will be converted to lower case.

**uppercase**

All characters in the resulting property extract will be converted to upper case.

**uxTimeStamp**

This is a special switch for date conversions. It only works if the extracted property value is an ISO-like timestamp (YYYY-MM-DD HH:MM:SS). If so, it will be converted to a Unix-like ctime() timestamp. If the extracted property value is not an ISO-like timestamp, no conversion happens.

**uxLocalTimeStamp**

This is the same as uxTimeStamp, but with local time instead of GMT.

**date-rfc3339**

This option is for replacing the normal date format with the date format from RFC3339.

**date-rfc3164**

This option is for replacing the normal date format with the date format from RFC3164.

**date-rfc3164strict**

Does the same as date-rfc3164 but when the date is below 10, two spaces will be added between Month and day (Which is defined in rfc3164).

**escapecc**

Control characters* in property are replaced by the sequence `##hex-val##`, where* hex-val is the hexadecimal value of the control character (at least two digits, may be more).

**spacecc**

Control characters* in the property are replaced by spaces. This option is most* useful when a message contains control characters (e.g. a Windows Event Log Message) and should be written to a log file.

**compressspace**

Compresses multiple consecutive space characters into a single one. The result is a string where all words are separated by just single spaces. To also compress control characters, use the compressspace and spacecc options together (e.g.``'%msg:::spacecc,compressspace%'). Please note that space compression happens on the final substring. So if you
use the FromPos and ToPos capabilities the substring is extracted first and
then the space compression applied. For example, you may have the msg string"1 2". There are two space between 1 and 2. Thus, the property replacer expression:``%msg:1:3:compressspace%

will lead to `"1  "` ('1' followed by two spaces). If you intend to receive
`"1 2"``('1' followed by one space, followed by '2'), you need to use ``%msg:1:4:compressspace%`

or

`%msg:1:/2/$:compressspace%`

In the second case, the exact length of the uncompressed string is not known, thus a search is used in topos to obtain it. The result is then space-compressed.

**compsp**

Exactly the same as compressspace, just an abbreviated form for those that like it brief.

**csv**

For example `%variable:::csv%`. This option will create a valid CSV string. For example a string like this:`this is a "test"!` becomes this `"this is a ""test""!"` where quotes are replaced with double quotes.

**cef**

Convert string content into valid McAfee CEF Format. This means that `=``will be replaced with=``` and `\` will be replaced with `\`. **convgermuml**

Converts German Umlaut characters to their official replacement sequence (e.g. "ö" –> "oe")

**localtime**

Now you can print the Time with localtime format by using``%variable:::localtime%``

**nomatchblank**

If this is used, the Property Replacer will return an empty string if the frompos or topos is not found.

**replacepercent**

This option replaces all `%` occurrences with a double `%%`, which is needed for the property replacer engine in case that a string is reprocessed. This is needed because the percent sign is a special character for the property replacer. Once the property is processed, the double``%%``become automatically one``%``. **toipv4address**

Property string will be converted into IPv4 Address format if possible.

**toipv6address**

Property string will be converted into IPv6 Address format if possible.

**crlftovbar**

Does the same as date-rfc3164 but when the date is below 10, two spaces will be added between Month and day (Which is defined in rfc3164).

**removecc**

Removes all control characters from 0x00 to 0x1F

**replacechar**

Replaces a single character with another single character.

How ASCII characters are being handled:

Sample:`%msg:$x:$y:replacechar%`

Broken down:

`%msg:$``<- Tells property replacer that a character is being expected (At the moment only for REPLACECHAR Option). ``x``<- The character to search for ``:`

`$``<- Tells property replacer that a character is being expected (At the moment only for REPLACECHAR Option). ``y``<- The character to replace with ``:`

`replacechar%`

How special characters are handled?

Sample: `%msg:$\n:$|:replacechar%`

`%msg:$` <- Tells property replacer that a character is being expected (At the moment only for REPLACECHAR Option). `\n` <- The character to search for special character, possible values: t for tab,

n for newline,

v for verticaltab,

f for formfeed,

r for carriage return

for an actual backslash.``:``

`$``<- Tells property replacer that a character is being expected (At the moment only for REPLACECHAR Option). ``|``<- The character to replace with ``:`

`replacechar%`

**minlength=N**

Pads the transformed output on the right with spaces until it contains at least `N` characters. For example, `%msg:1:3:minlength=10%` emits the first three message characters followed by seven spaces. Options compose: `%msg:1:3:uppercase,minlength=10%`.

* = control characters like e.g. carriage return, line feed, tab, …*

**Important:** All option values are case-sensitive. So “uxTimeStamp” works while “uxtimestamp” is an invalid option!

### Simple Examples

A good example for this is the email subject line, which has severe length constraints. If you would like to have only the first 40 characters of the actual message text in the subject, you could use the replacer: `"%msg:1:40%"`. If you know the first 10 characters of the message are meaningless for you but you would like to see the full rest of the message (no matter how long it may be), you can use a sequence like `"%msg:11%"`.

If you would just like to see the plain message from beginning to end, you can simply omit frompos and topos: `"%msg"`. Of course, all of these sample not only work with the “msg” property, but also with all others like “facility”, or “priority”, or W3C-log header extracted property names.

**More complex Examples**

If you would like to extract the 50 characters from the message after the word DROP, you would use the following replacer string:`%msg:/DROP/$:+50%`

If you would like to have the first 40 characters in front of the string “- aborted” (including that string):

`%msg:/- aborted/$:-40%`

If you would like to receive everything starting from (and including) “Log:”:

`%msg:/Log/%`

If you would like to have everything between the string “FROM” and “TO” including NONE of the both searchstrings:

`%msg:/FROM/$:/TO/%`

If you would just like to log lowercase letters in your log messages:

```
%msg:::lowercase%
```

And if you would just like to have the first 50 characters (and these in lower case):

```
%msg:50:::lowercase%
```

If you need to change a timestamp to a UNIX-like timestamp, you could use this:

```
%datereceived:::uxTimeStamp%
```

Please see also the focused sample in the topos description.

**A real world Sample**

We use the following template to generate output suitable as input for MoniLog:

```
%timegenerated:1:10%,%timegenerated:12:19%,%source%,%syslogfacility%,%syslogprio rity%,EvntSlog
: %severity% %timereported:::uxTimeStamp%: %source%/%sourceproc% (%id%) - "%msg%"%$CRLF%
```

**Please note: everything is on one line with no line breaks in between. This** example is from the “write to file” action (with custom file format).**

### System Properties

System properties are special sequences that can be helpful. They are available with all event types. They are:

**$CRLF**

A Windows newline sequence consisting in the characters CR and LF. If you just need CR, you can use `%$CRLF:1:1%` and if you need use LF you can use `%$CRLF:2:2%`

**$TAB**

An US-ASCII horizontal tab (HT, 0x09) character

**$HT**

same as $TAB

**$CR**

A single US-ASCII CR character (shortcut for `%$CRLF:1:1%`) **$LF**

A single US-ASCII LF character (shortcut for``%$CRLF:2:2%``) **$xNN**

A single character, whose value (in hexadecimal) is given by NN. NN must be two hexadecimal digits - a leading zero must be used if a value below 16 is to be represented. The value 0 (%x00) is invalid and - if specified - replaced by the “?” character.

As an example, $CR could also be expressed as %$x0d%.

Please note that only one character can be represented. If you need to specify multiple characters, you need multiple $xNN sequences. An example may be $CRLF which could also be specified as %$x0d%%$x0a% (but not as %$x0d0a%).

**$NOW**

Contains the current date and time in the format:`YYYY-MM-DD HH.MM.SS`

Please note that the time parts are delimited by `'.'` instead of `':'`. This makes the generated name directly suitable for file name generation.

If you need just parts of the timestamp, please use the property replacer’s substring functionality to obtain the desired part. Use``%$NOW:1:4% to get the year,``

`%$NOW:6:7% to get the month,`

…

`%$NOW:1:10% to get the full datestamp,`

`%$NOW:12:20% to get the full timestamp`

**$NEWUUID**

Creates a new UUID (Universally Unique Identifiers), a unique 128-bit integer represented as a 32 digit hexadecimal number.

### Custom Properties

Users can create an unlimited number of custom properties. These can be created with for example the “PostProcess” action (if the product edition purchased supports this action).

Custom properties can theoretically have any name, but Adiscon highly recommends to prefix them with “u-” (e.g. “u-MyProperty” - “u” like “user”). This ensures that no compatibility problems will arise in current and future versions of the software. Adiscon guarantees that it will never use the “u-” prefix for Adiscon-assigned properties.

Custom properties can be used just like regular properties. Wherever you can specify a property, you can also specify a custom property.

### Event-Specific Properties

Each network event is represented by a so-called “Event Record” (sometime also named an “InfoUnit”, an “Unit of Information”). Data obtained from all services will end up as an event. For example, Windows Event Log data, syslog data, and a file line obtained by the file monitor will all be an event. That kind of generalization make it easy to deal with all of these events in a consistent way.

Each event has a set of properties which in turn have values. For example, there is a property named “source” and it will always contain an indication of which system the event originated on. Obviously, not every event source does support all properties. For example, a syslog message does not contain a Windows Event ID - simply because there is no such thing as an event ID in syslog. So, depending on the type of event, it may contain different properties.

In order to make the product really generally useful, some few properties have been defined in a generic way and are guaranteed to be present in every event, no matter what type it may have. Sometimes this is a “natural” common property, like the “fromhost”. Sometimes, though, it may look a bit artificial. An example of the later is the “syslogfacility” property. It is guaranteed to be present in every event - but actually this is a syslog-only thing. The non- syslog event sources either emulate this property (in a consistent manner) or allow the user to configure a syslogfacility that should be used for all events generated by that service. At the bottom line, this will ensure that the property is available in all events and - given proper configuration - that can be extremely helpful for the administrators to set up things in a powerful and generic way.

### Standard Properties

As outlined under Event Properties, these are properties present in all types of events. Some event types have only these standard properties. Others have additional properties. Those with additional properties are documented in the other sections. If there is no specific documentation for a specific event type, this means that it supports the standard properties, only.

**msgPropertyDescribed**

A human-readable representation of the message text. While this is generally available, the exact contents largely depends on the source of the information. For example, for a file monitor it contains the file line and for a syslog message it contains the parsed part of the syslog message.

**source**

The source system the message originated from. This can be in various representations (e.g. IP address or DNS name) depending on configuration settings.

**localhostname**

On service startup it is automatically set to the local system computer name. It is read only and can be used if source property is not usable. E.g. if the Source property cannot be translated to IP format because the event log entry was recorded with an old computer name that no longer exists.

**resource**

A user-assigned numerical value. Does not have any specific meaning. Primarily intended for quick filtering.

**CustomerID**

A user-assigned numerical value. Does not have any specific meaning. Primarily intended for quick filtering.

**SystemID**

A user-assigned numerical value. Does not have any specific meaning. Primarily intended for quick filtering.

**timereported**

The time the originator tells us when this message was reported. For example, for syslog this is the timestamp from the syslog message (if not configured otherwise). Please note that timereported eventually is incorrect or inconsistent with local system time - as it depends on external devices, which may not be properly synchronized.

For Windows Event Log events, timereported contains the timestamp from the event log record.

**timegenerated**

The time the event was recorded by the service. If messages are forwarded via SETP, this timestamp remains intact.

**importance**

Reserved for future use.

**iut**

Indicates the type of the event. Possible values are:

```
1- syslog message
2- heartbeat
3- Windows Event Log Entry
4- SNMP trap message
5- file monitor
8- ping probe
9- port probe
10- Windows service monitor
11- disk space monitor
12- database monitor
13- serial device monitor
```

**iuvers**

Version of the event record (info unit). This is a monitorware internal version identifier.

### Windows Event Log Properties

**id**

Windows Event ID

**severity**

severity as indicated in the event log. This is represented in string form. Possible values are:

```
[INF] - informational
[AUS] - Audit Success
[AUF] - Audit failure
[WRN] - Warning
[ERR] - Error
[NON] - Success (called "NON" for historical reasons)
```

**severityid**

The severity encoded as a numerical entity (like in Windows API)

**sourceproc**

The process that wrote the event record (called “source” in Windows event viewer).

**category**

The category ID from the Windows Event Log record. This is a numerical value. The actual value is depending on the event source.

**catname**

The category name from the Windows Event Log record. This is a string value. The actual value is depending on the event source. This value is a textual representation from the Category ID. This property could contain line feeds, which can be removed by activating the option “Remove Control Characters from String Parameters” in the advanced options of the EventLog Monitor Service.

**user**

The user name that was recorded in the Windows Event Log. This is “NA” if no user was recorded.

**NTEventLogType**

The name of the Windows Event Log this event is from (for example “System” or “Security”).

**bdata**

Windows Event Log records sometimes contain binary data. The Event Log Monitor service can be set to include this binary data into the event, if it is present. If it is configured to do so, the binary data is put into the “bdata” property. Every byte of binary data is represented by two hexadecimal characters.

Please note that it is likely for bdata not to be present. This is because the binary data is seldom used and very performance-intense. (%id%) - “%msg%”%$CRLF%

### Windows Event Log V2 Properties

**id**

Windows Event ID

**severity**

severity as indicated in the event log. This is represented in string form. Possible values are:

```
[INF] - informational
[AUS] - Audit Success
[AUF] - Audit failure
[WRN] - Warning
[ERR] - Error
[NON] - Success (called "NON" for historical reasons)
```

**severityid**

The severity encoded as a numerical entity (like in Windows API)

**sourceproc**

The process that wrote the event record (called “source” in Windows event viewer).

**category**

The category ID from the Windows Event Log record. This is a numerical value. The actual value is depending on the event source.

**catname**

The category name from the Windows Event Log record. This is a string value. The actual value is depending on the event source. This value is a textual representation from the Category ID. This property could contain line feeds, which can be removed by activating the option “Remove Control Characters from String Parameters” in the advanced options of the EventLog Monitor Service.

**user**

The user name that was recorded in the Windows Event Log. This is “NA” if no user was recorded.

**nteventlogtype**

The name of the Windows Event Log this event is from (for example “System” or “Security”).

**channel**

The channel property for event log entries, for classic Event logs they match the `%nteventlogtype%` property, for new event logs, they match the “Event Channel”.

**sourceraw**

This contains the full internal name of the event source for new event logs, for classic event logs it contains the same value as in %sourceproc%.

**level**

Textual representation of the event log level (which is stored as a number in `%severityid%`). This property is automatically localized by the system.

**categoryid**

Internal category id as number.

**keyword**

Textual representation of the event keyword. This property is automatically localized by the system.

**user_sid**

If available, contains the raw SID of the username (`%user%`) property.

**recordnum**

Contains the internal event record number. Please note that if the event log has been truncated before, it may not start with 0 or 1 but a higher number.

### Syslog Message Properties

**rawsyslogmsg**

The message as it was received from the wire (unparsed).

**syslogfacility**

The facility of a syslog message. For non-syslog messages, the value is provided based on configuration. In essence, this is simply an integer value that can be used for quick filtering inside your rules.

**syslogfacility_text**

The facility of a syslog message. This property is automatically created by using the syslogfacility properly and set to these values: `"Kernel", "User", "Mail", "Daemons", "Auth", "Syslog", "Lpr", "News", "UUCP", "Cron", "System0", "System1", "System2", "System3", "System4", "System5", "Local0", "Local1", "Local2", "Local3", "Local4", "Local5", "Local6", "Local7"`

**syslogpriority**

The severity of a syslog message. For non-syslog messages, this should be a close approximation to what a syslog severity code means.

**syslogpriority_text**

The severity of a syslog message. This property is automatically created by using the syslogpriority properly and set to these values:
`"Emergency", "Alert", "Critical", "Error", "Warning", "Notice", "Informational", "Debug"`

**syslogtag**

The syslog tag value, a short string. For non-syslog messages, this is provided based on configuration. In most cases, this is used for filtering.

**syslogver**

Contains the syslog version number which will be one or higher if a rfc 5424 valid message has been received, or 0 otherwise

**syslogappname**

Contains the appname header field, only available if the Syslog message was in rfc 5424 format. Otherwise, this field will be emulated by the %syslogtag% property

**syslogprocid**

Contains the procid header field, only set if the Syslog message was in rfc 5424 format.

**syslogmsgid**

Contains the msgid header field, only set if the Syslog message was in rfc 5424 format.

**syslogstructdata**

Contains the structdata header field (in raw format), only set if the Syslog message was in rfc 5424 format.

**syslogprifac**

Contains combined syslog facility and priority useful to build your own custom syslog headers

### Disk Space Monitor

**currusage**

The currently used disk space.

**maxavailable**

The overall capacity of the (logical) disk drive.

### CPU/Memory Monitor

**wmi_type**

This variable is a string and can be one of the following variables: cpu_usage, mem_virtual_usage, mem_physical_usage, mem_total_usage.

**cpu_number**

Number of the current checked CPU.

**cpu_load**

The workload of the CPU as number, can be 0 to 100.

**mem_virtual_load**

How much virtual memory is used (MB).

**mem_virtual_max**

How much virtual memory is max available (MB).

**mem_virtual_free**

How much virtual memory is free (MB).

**mem_physical_load**

How much physical memory is used (MB).

**mem_physical_max**

How much physical memory is max available (MB).

**mem_physical_free**

How much physical memory is free (MB).

**mem_total_load**

How much total(Virtual+Physical) memory is used (MB).

**mem_total_max**

How much total(Virtual+Physical) memory is max available (MB).

**mem_total_free**

How much total(Virtual+Physical) memory is free (MB).

### File Monitor

**genericfilename**

The configured generic name of the file being reported.

**generatedbasefilename**

Contains the generated file name without the full path.

### Special IIS LogFile Properties

The Logfile Fields in IIS Logfiles are customizable, so there is no hardcoded command for their use.

The property-name depends on its name in the logfile. For example we take this Logfile:

```
#Software: Microsoft Internet Information Services 5.0
#Version: 1.0
#Date: 2005-10-27 14:15:25
#Fields: date time c-ip cs-username s-ip s-port cs-method cs-uri-stem
cs-uri-query sc-status cs(User-Agent)
2005-10-27 14:15:16 127.0.0.1 - 192.168.0.1 443 POST /eCommerce/asdf.php
2005-10-27 14:15:16 127.0.0.1 - 192.168.0.1 443 POST /eCommerce/asdf.php
2005-10-27 14:15:16 127.0.0.2 - 192.168.0.1 443 POST /eCommerce/asdf.php
2005-10-27 14:15:16 127.0.0.2 - 192.168.0.1 443 POST /eCommerce/asdf.php
```

As you can see, in our sample the fields are named: date, time, c-ip, cs-username, s-ip, and so on.

To use them as a Property inside our MonitorWareProducts, just use the names from your Logfile and add a “p-” before it:

**p-date**

The Date on which the Event occurs

**p-time**

The Time on which the Event occurs

**p-c-ip**

The IP address of the User which accessed

**p-cs-username**

The Username of the User which accessed

**p-s-ip**

The Server IP

**p-s-port**

The Server Port

**p-cs-method**

The Client-Server Method (POST,GET)

**p-cs-uri-stem**

The accessed File including its path

### Windows Service Monitor

**sourceproc**

The name of the service whose status is being reported (from the Windows service registry).

### Ping Probe

**echostatus**

Status returned for the echo request

The status value can be one of the following:

```
0 = IP_SUCCESS
11002 = IP_DEST_NET_UNREACHABLE
11003 = IP_DEST_HOST_UNREACHABLE
11010 = IP_REQ_TIMED_OUT
11013 = IP_TTL_EXPIRED_TRANSIT
11016 = IP_SOURCE_QUENCH
11018 = IP_BAD_DESTINATION
```

**roundtriptime**

Round trip time for the ping packet (if successful)

### Port Probe

**responsestatus**

The status of the probe.

**responsemsg**

The response message received (if any)

### Database Monitor

Database-Monitor created events are a bit different than other events. The reason is that the database fields themselves become properties - but obviously these are not fixed but depend on what you monitor.

All queried data fields are available as properties via their database field name **prefixed with “db-”**.

An example to clarify: we assume the following select statement is used for the database monitor:

select name, street, zip, city from addresses

There is also an ID column named “ID”. So the event generated by this database monitor will have the following specific properties:

- db-ID
- db-name
- db-street
- db-zip
- db-city

These properties will contain the field values as they are stored in the database. Please note that NULL values are translated into empty strings (“”), so there is no way to differentiate a NULL value from an empty string with this version of the database monitor.

Other than the custom “db-” properties, no specific database monitor properties exist.

### Serial Monitor

**portname**

The name of the port that the data originated from (typical examples are COM1, COM2). The actual name is taken from the configuration settings (case is also taken from there).

### MonitorWare Echo Request

**responsestatus**

The status of the echo request. Possible values:

```
0 - request failed (probed system not alive)
1 - request succeeded
```

If the request failed, additional information can be found in the * msg* standard property.

### FTP Probe

**ftpstatus**

The status of the connection.

**ftprespmsg**

The response of the connection.

### IMAP Probe

**imapstatus**

The status of the connection.

**imaprespmsg**

The response of the connection.

### NNTP Probe

**nntpstatus**

The status of the connection.

**nntprespmsg**

The response of the connection.

### SMTP Probe

**smtpstatus**

The status of the connection.

**smtprespmsg**

The response of the connection.

### POP3 Probe

**pop3status**

The status of the connection.

**pop3respmsg**

The response of the connection.

### HTTP Probe

**httpstatus**

The status of the connection.

**httprespmsg**

The response of the connection.

## Complex Filter Conditions

The rule engine uses complex filter conditions.

Powerful boolean operations can be used to build filters as complex as needed. A boolean expression tree is graphically created. The configuration program is modeled after Microsoft Network Monitor. So thankfully, many administrators are already used to this type of Interface. If you are not familiar with it, however, it looks a bit confusing at first. In this chapter, we are providing some samples of how boolean expressions can be brought into the tree.

### Example 1

In this example, the message text itself shall be checked. If it contains at least one of three given strings, the filter should become true. If none of the string is found, the boolean expression tree evaluates to false, which means the associated action(s) will not be executed.

In pseudo-code, the filter could be written like this:

```
If (msg  = "DUPADDRES") OR (msg = "SPANTREE") OR (msg = "DUPLEX_MISMATCH) then
    execute action(s)
end if
```

Please note: in the example, we have abbreviated “message” to just “msg”. Also note that for brevity reasons we use the equals (“=”) comparison operator, not the contains. The difference between the equals and the contains operator is that with “contains”, the string must just be part of the message.

In the filter dialog, this pseudo code looks as follows:

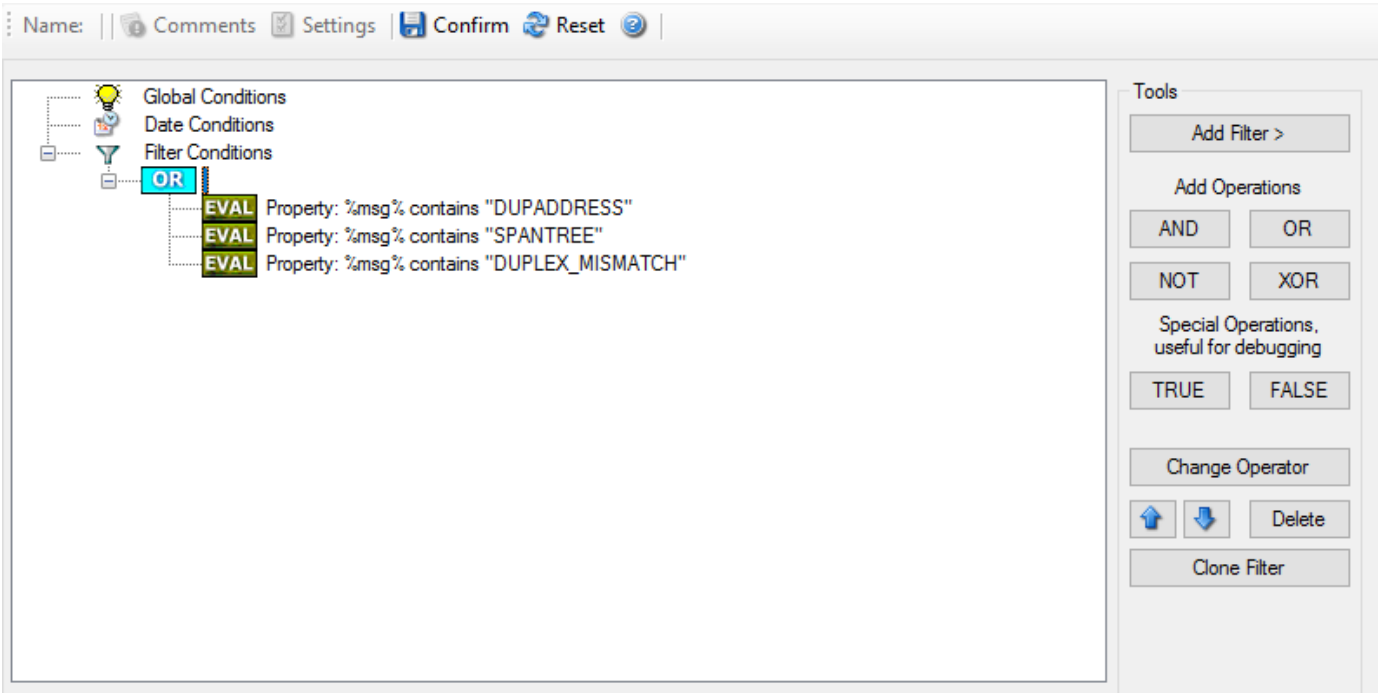



Figure 1 - Example 1

### Example 2

Example 2 is very similar to example 1. Again, the message content is to be checked for three string. This time, all of these strings must be present in order for the boolean tree to evaluate to false.

The pseudo code would be as follows (under the same conditions outlined in example 1 above):

```
If (msg  = "DUPADDRES") AND (msg = "SPANTREE") AND (msg = "DUPLEX_MISMATCH) then
    execute action(s)
end if
```

In the filter dialog, this pseudo code looks as follows:

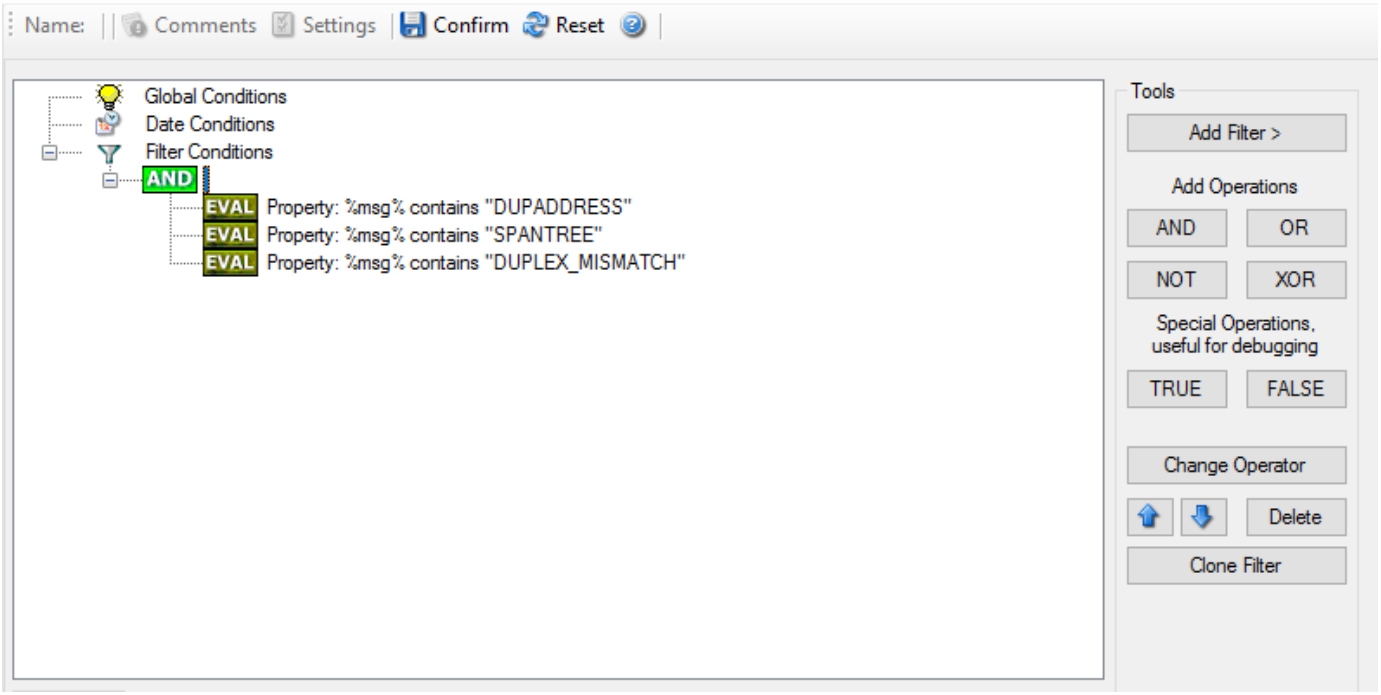



Figure 2 - Example 2

### Example 3

This example is a bit more complex version of example 1. Again, the same message text filtering is done, that is if any one of the provided substrings is present, the filter eventually evaluates to true. To do so, the source system must also contain the string “192.0.2”, which can be used to filter on a device from a specific subnet.

An example like this can be used for a rule where the administrator of a specific subnet should be emailed when one of the strings indicate a specific event.

The pseudo code would be as follows (under the same conditions outlined in example 1 above):

```
If ((sourceSys = "192.0.2") And
        ((msg  = "DUPADDRES") OR (msg = "SPANTREE")
            OR (msg = "DUPLEX_MISMATCH))) then
    execute action(s)
end if
```

In the filter dialog, this pseudo code looks as follows:

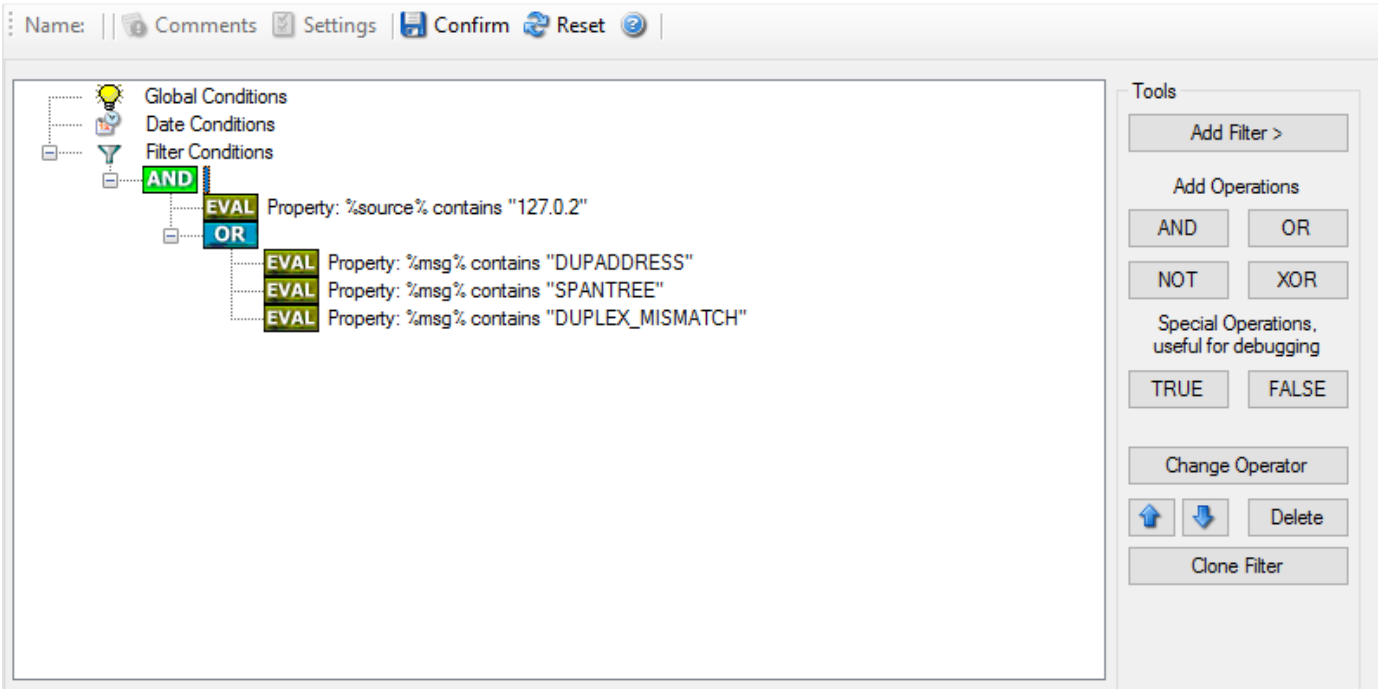



Figure 3 - Example 3

As a side note, you may want to use a range check instead of a simple include for the source system. With a range string check, you can specify that the string must be within a specified column range, in this case obviously at the beginning of the source system IP address.

### Real-World Examples

To see some real-world examples of where boolean conditions inside filtering are used, please visit these web links:

- Detecting Password Attacks under Windows

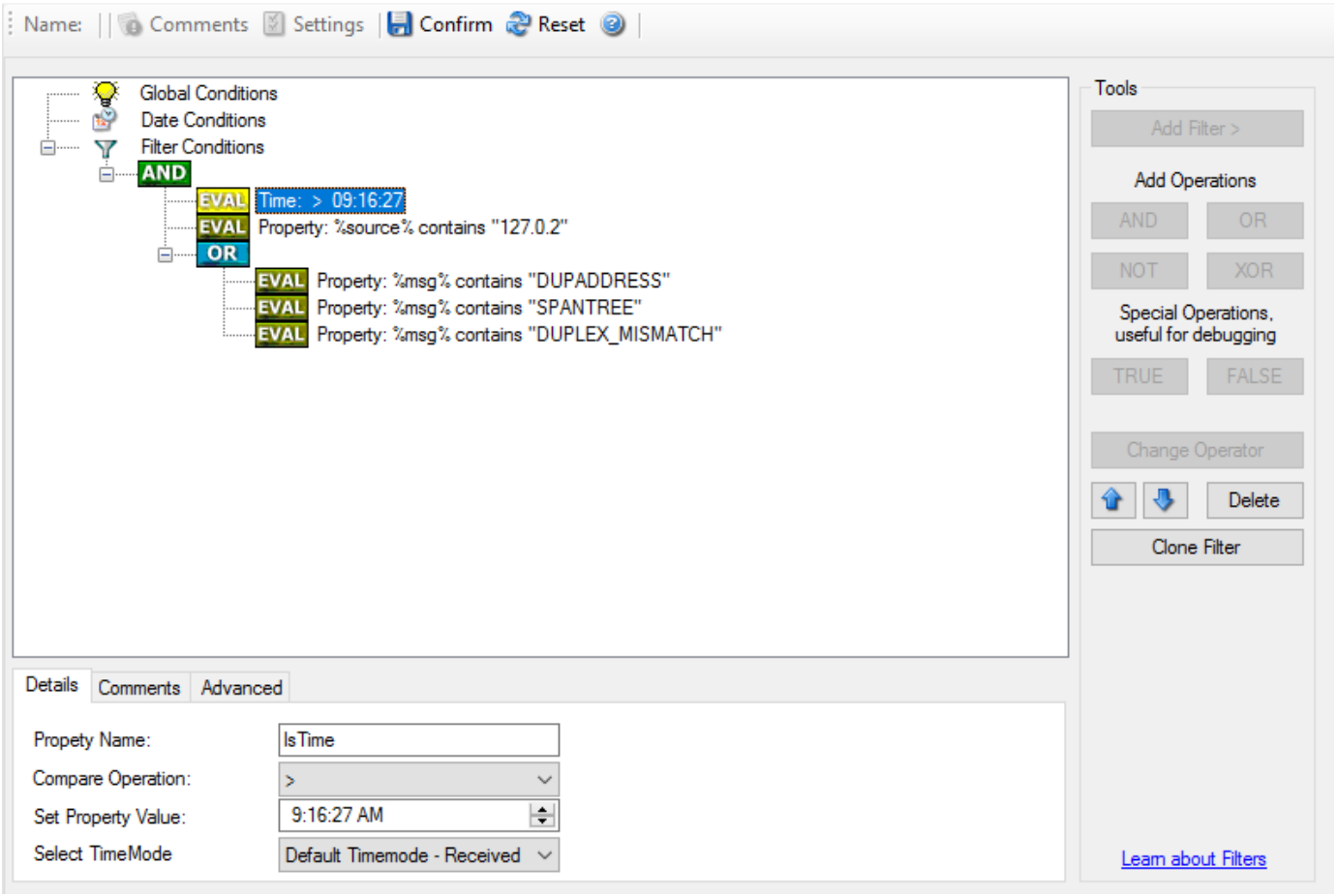
### Example 4

In this example, the report is to be filtered in such a way that it shows information only in the case, if the time is greater then certain time with certain event source and one of two event ID's.

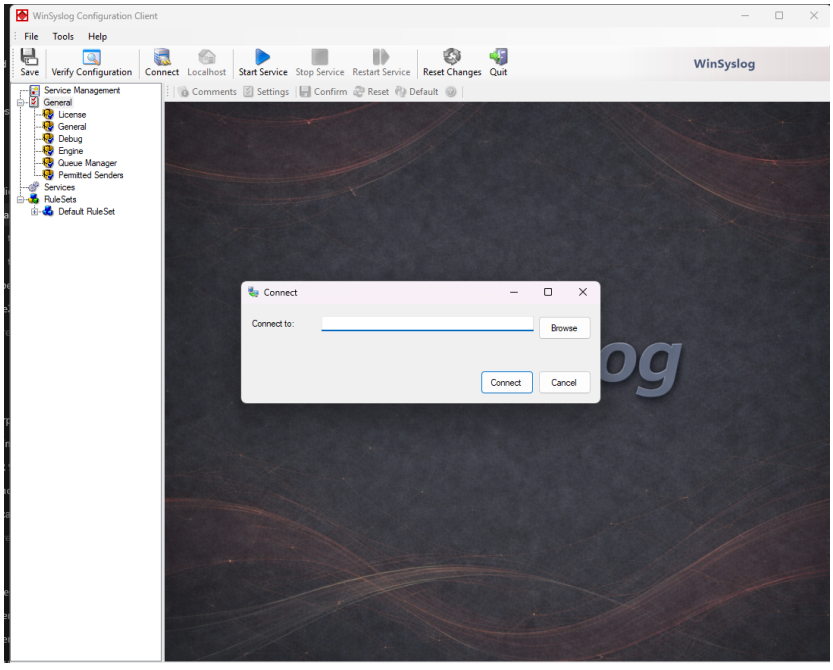
In pseudo-code, the filter could be written like this:

```
If (DeviceReportedTime is greater than {9:16:27} AND EventSource is equal to
{Print} AND [EventID is equal to {10} OR EventID is equal to {18}]]
```

In the filter dialog, this pseudo code looks as follows:

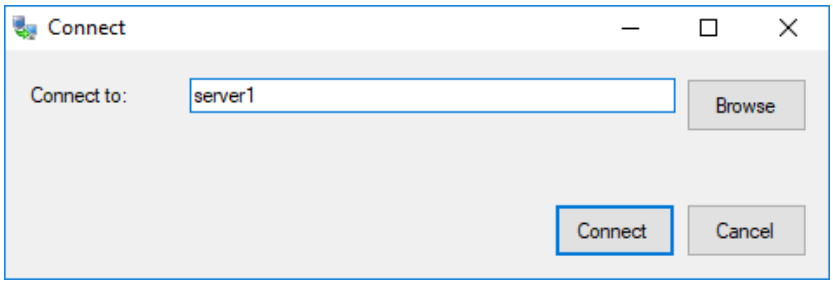



## Connect to Computer




*WinSyslog Connect to Computer window*

Click the Connect button in order to access another machine remotely. A window will open up.




Here you can enter the name of the machine you want to configure remotely. You can either directly enter the name into the textfield or you use the Browse button to see a list of available machines in the network. The click on the Connect button, the configuration client will verify access to the remote machine. If the verification is successful, you will be able to proceed with the remote access. Otherwise an error message will be shown.

**Please Note:** For remote configurations, you must ensure, that the remote machine

is accessible by network and has access rights for the current logged on local user.

## Registry Paths

Here are some more details regarding registry paths.

Since 64bit Windows re-routes all registry keys for 32bit programs automatically (`HKEY_LOCAL_MACHINE\SOFTWARE\ to HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node`), Adiscon decided in the course of development that the 32bit as well as 64bit registry keys of the service should use the``„HKEY_LOCAL_MACHINESOFTWAREWow6432Node“`` subkey.

But for your case, this is rather problematic since you need a registry file for 32bit and for 64bit systems. Thus we decided to use a workaround, namely to use the parameters key in the service area. In the services subkey of the registry is no automatic mapping for Win32/64 applications and thus you can use the same registry file for all systems.

## System Error Codes

In most of the cases where you will get system error codes, see this list for their meanings: System Error Codes

If you cannot find them there, please contact us via the Customer Service System.

## User Interface changes

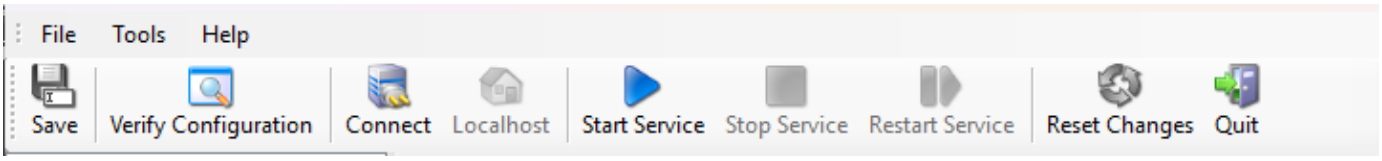
The **26.07** installer includes the established configuration client and the new user interface for this product:

- **Established configuration client** - the WinForms client documented in this manual and the primary reference for setup steps and screenshots.
- **WinSyslog User Interface** - the newer WinUI-based interface (`adiscon-client-ng`) available alongside the established configuration client.

The established configuration client supports license file application and full configuration editing. See your product website's **Upgrade** article for the customer-facing description of the product-specific user interface.

### Property window toolbar

When you open a service or action property window, the toolbar provides **Save**, **Verify Configuration**, **Connect**, **Start**, **Stop**, **Restart**, **Reset Changes**, and **Quit**. **Reset Changes** discards unsaved edits on the open form only; it does not restore factory defaults for the selected item.

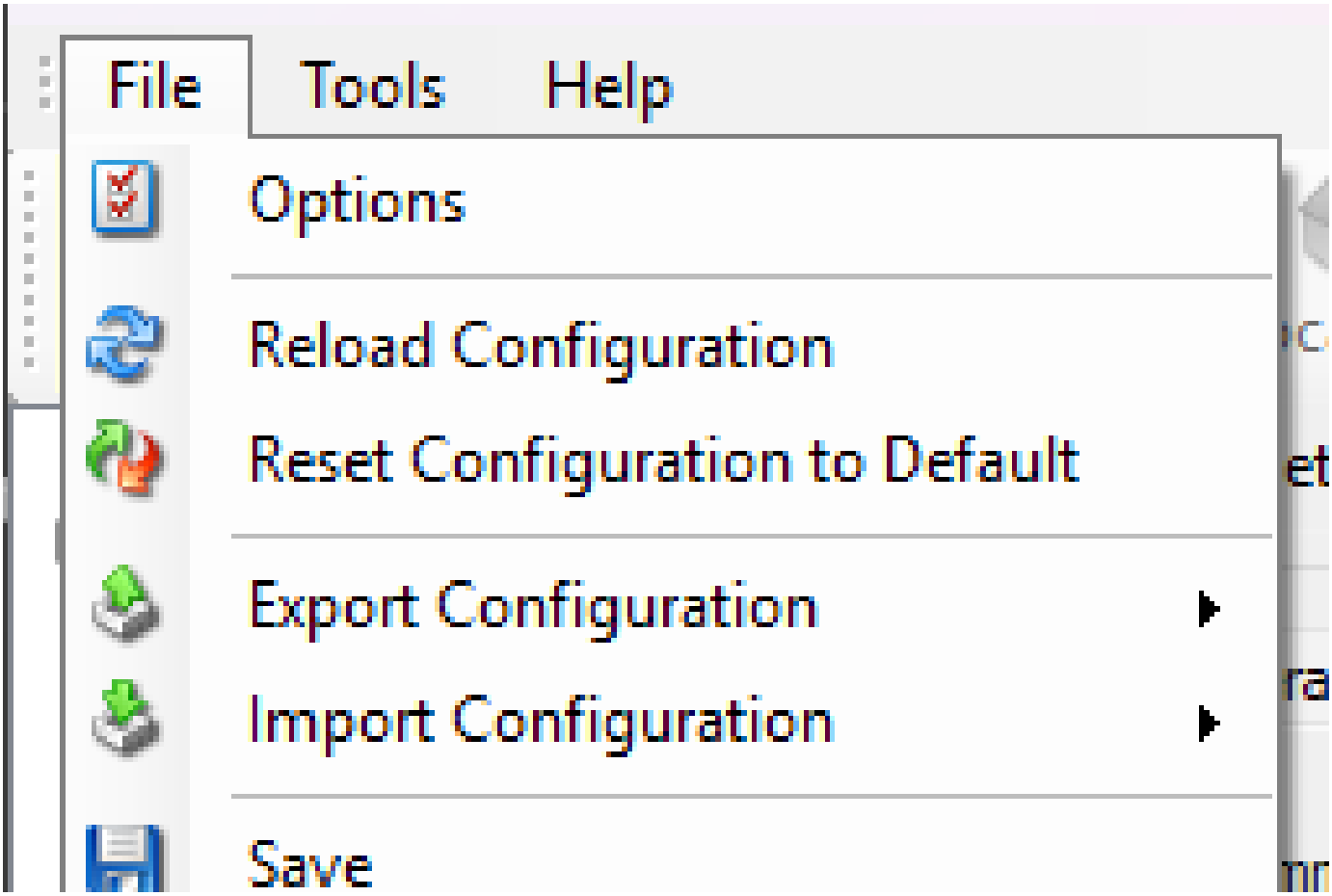



The **main window** (rules tree) has a separate toolbar with **Start**, **Stop**, **Restart**, **Kill**, and **Open Windows Services** for the background service. **Kill** ends an orphan process when the service manager state is inconsistent. **Open Windows Services** opens `services.msc` for Windows Service management.

When the service is **paused** during configuration reload, stop and restart may be disabled until reload completes.

### Reset configuration to default

The **File** menu provides **Reset Configuration to Default**, which resets the whole deployed configuration:



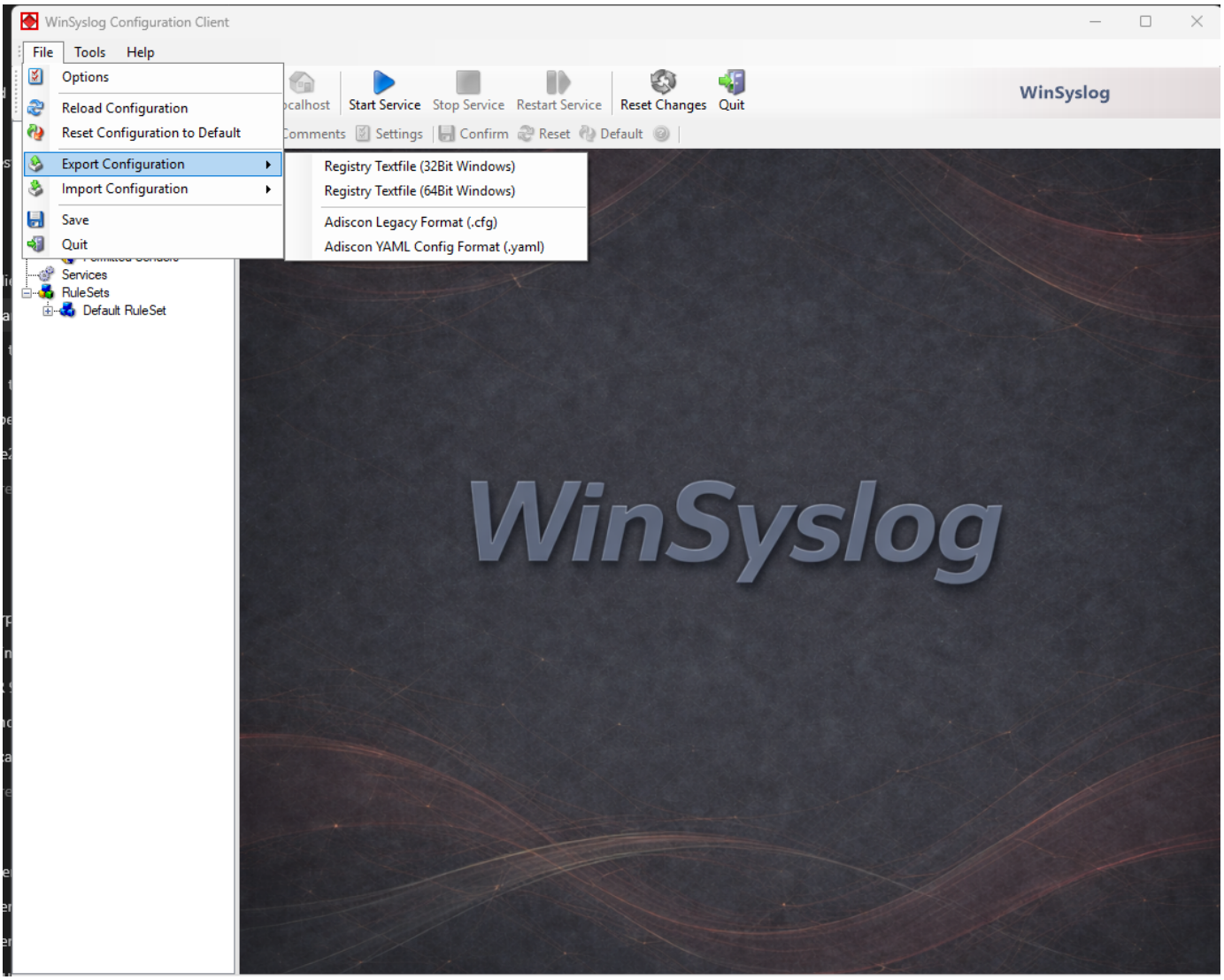

This is separate from **Reset Changes** on an open property form. It is also separate from the context menu or toolbar **Reset** command for a selected service or action, which restores only that selected item to factory defaults.

### Event Viewer

After start, stop, or configuration reload, service-management events are prepended in the Event Viewer so recent control actions appear at the top.

### File formats

The **File** menu supports:




*WinSyslog File menu with YAML import and export commands*

- **Adiscon YAML Config Format** (`.yaml`) - import and export
- **Adiscon Legacy Format** (`.cfg`) - legacy text configuration

See the file-based configuration chapter in your product manual and What happens when the client opens a configuration with unsupported blocks?.

### Related information

- Why did product versions jump to 26.07?

## Glossary

This section defines the main terms, protocols, and reference concepts that appear throughout the WinSyslog manual. Unlike the Concepts section, this glossary is meant for term lookup rather than for explaining the overall product model.

### Engine Only Install

An **Engine Only Install** refers to a deployment method where only the core service executable (e.g., `winsyslg.exe`, `mwagent.exe`, `evtlog.exe`) and its essential dependencies are installed, without the full client application and user interface components. This type of installation is particularly useful for:

- **Mass deployments** where you want to minimize the installation footprint
- **Server environments** where GUI components are not needed
- **Automated deployments** where configuration is managed via registry files
- **Security-conscious environments** where you want to reduce the attack surface

In an engine-only install, the service runs with the configuration stored in the Windows Registry, which can be exported from a master installation and imported during deployment. This allows for consistent configuration across multiple systems without requiring the full installation package.

Key characteristics: - Smaller disk footprint (typically just a few MB) - No GUI components or client tools - Configuration via registry import/export - Suitable for headless/server deployments - Can be updated by simply replacing the executable files

This approach is commonly used in enterprise environments where hundreds or thousands of systems need to be monitored with consistent configuration.

### FTP

FTP stands for File Transfer Protocol. FTP is the best means for moving large files across the Internet. FTP is a client/server protocol that enables a user with an FTP client to log on to a remote machine, navigate the file system of that remote machine, and upload and download files from that machine.

There are two basic types of FTP on the Internet anonymous ftp and private ftp. With anonymous ftp, one logs in as user anonymous, giving one’s email address as a password. With private FTP, one logs in with the username and password one has established on that particular system. You are logged into your home directory, with all the file permissions you would normally have there.

### HTTP

HyperText Transfer Protocol, the underlying protocol used by the World Wide Web. HTTP defines how messages are formatted and transmitted, and what action Web servers and browsers should take in response to various commands. For example, when you enter a URL in your browser, this actually sends an HTTP command to the Web server directing it to fetch and transmit the requested Web page.

### IETF

The IETF is an important Internet standards body. IETF is a short name for “Internet Engineering Task Force”. The IETF is responsible for the creation of RFCs. Unlike other, formal standards bodies it is loosely organized. There is no specific membership to the IETF, anyone (knowledgeable) can become an IETF member just by participating on the IETF discussion mailing lists.

The IETF itself provides a good overview over itself at https://www.ietf.org/about/mission/.

### IMAP

Internet Message Access Protocol. A protocol allowing a client to access and manipulate electronic mail messages on a server. It permits manipulation of remote message folders (mailboxes), in a way that is functionally equivalent to local mailboxes.

IMAP includes operations for creating, deleting, and renaming mailboxes; checking for new messages; permanently removing messages; searching; and selective fetching of message attributes, texts, and portions thereof. It does not specify a means of posting mail; this function is handled by a mail transfer protocol such as SMTP.

### IPv6

Adiscon Products officially support IPv6. The IPv6 support was introduced with the following versions:

- WinSyslog 8.0
- WinSyslog 11.0
- EventReporter 12.0

Support for IPv6 is available in all network related facilities of the engine. All network related actions will automatically detect IPv6 and IPv4 target addresses if configured. You can also use DNS resolution to resolve valid IPv6 addresses. Network related Services can either use IPv4 or IPv6 as internet protocol. In order to support both protocols, you will need to create two services. The only exception is the RELP Listener, which uses IPv4 and IPv6 automatically if available.

### Millisecond

A millisecond is a thousand of a second. It is abbreviated as “ms”. As such, 500ms mean half a second.

Inside the **`adiscon’s monitorware line of products`**, many timers are expressed in milliseconds as a fine control over the services and actions is provided to the administrator.

### NNTP

NNTP stands for Network News Transport Protocol. This protocol is used by client and server software to carry USENET postings back and forth over a TCP/IP network.

When you are using any of the common software like modern web browsers or newsreaders, you are taking benefit of the NNTP connection to participate in newsgroups.

### POP3

POP3 (Post Office Protocol 3) is the most recent version of a standard protocol for receiving e-mail. POP3 is a client/server protocol in which e-mail is received and held for you by your Internet server. Periodically, you (or your client e-mail receiver) check your mail-box on the server and download any mail. It is also built into modern web browsers and email clients.

### Registry File

A **Registry File** (with .reg extension) is a text file that contains a snapshot of Windows Registry entries. In the context of Adiscon products (WinSyslog, EventReporter, WinSyslog), registry files are used to export and import complete product configurations.

Registry files enable: - **Configuration backup** - Save your entire product configuration - **Mass deployment** - Apply the same configuration to multiple systems - **Configuration sharing** - Share configurations between team members - **Disaster recovery** - Quickly restore configurations after system failures

The registry file can be created through the product's client interface using the "Export Settings to Registry File" option, and imported silently using: `regedit.exe /s configuration.reg`

This makes registry files an essential tool for enterprise deployments and configuration management.

### RELP

RELP is the “Reliable Event Logging Protocol”. It assures that no message is lost in transit, not even when connections breaks and a peer becomes unavailable. The current version of the RELP protocol has a minimal window of opportunity for message duplication after a session has been broken due to network problems. In this case, a few messages may be duplicated (a problem that also exists with plain tcp syslog).

RELP addresses many shortcomings of the traditional plain tcp syslog protocol. For some insight into that, please have a look at https://rainer.gerhards.net/2008/04/on-unreliability-of-plain-tcp-syslog.html. Please note that RELP is currently a proprietary protocol. So the number of interoperable implementations is limited.

Note that for reliable operation where messages should be preserved over a service shutdown, queue cache mode must be activated.

### Resource ID

The Resource ID is an identifier used by the **`adiscon's monitorware line of products`**. It is a simple, administrator assigned string value. It can be used to correlate different events - even from different source - to a specific resource.

For example, on a Windows server running Microsoft Exchange, all Exchange events could be assigned to a resource id of “Exchange Server”.

In MonitorWare product page and WinSyslog support for Resource IDs is limited. The field is present and can be persisted to the database or stored in XML files, but besides this there is no value in it.

### RFC 3164

RFC 3164 is a IETF document. It describes how syslog messages have been seen in traditional implementations. RFC 3164 is not a standard but rather a descriptive (“informational” in IETF terms) document. It does not demand a specific behavior but rather documents what has been seen. Some existing implementations of real-world syslog use different formats.

RFC 3164 is just the first step towards a newer and better syslog standard. A standard already produced by this working group is rfc 3195, which describes how syslog can be sent reliably over a tcp connection.

Adiscon supports RFC 3164 messages. There are a number of switches in each product to take care of those implementation that do it slightly different.

The formal specification for RFC 3164 can be found in the IETF RFC repository.

### RFC 3195

RFC 3195 is an IETF standard. It specifies how syslog messages can reliably be transmitted via a tcp connection. RFC 3195 optionally allows for message encryption and authentication of sender and receiver. However, it has not receive any importance in practice. Servers are hard to find.

**`adiscon's monitorware line of products`** implement the core RFC 3195 protocol (actually, Adiscon was the first one to do this on the Windows platform). Under UNIX rsyslog and SDSC syslog are known to support RFC 3195. Our liblogging project enables your own applications to "talk" 3195.

The formal specification for RFC 3195 can be found in the IETF RFC repository .

During its creation, RFC 3195 was known as "syslog-reliable". Many people still use this name to refer to it.

### RFC 5424

RFC 5424 is a IETF document.

This document describes the syslog protocol, which is used to convey event notification messages. This protocol utilizes a layered architecture, which allows the use of any number of transport protocols for transmission of syslog messages. It also provides a message format that allows vendor-specific extensions to be provided in a structured way.

A standard already produced by this working group is rfc 3195, which describes how syslog can be sent reliably over a tcp connection.

Adiscon supports RFC 5424 messages. There are a number of switches in each product to take care of those implementation that do it slightly different.

The formal specification for RFC 5424 can be found in the IETF RFC repository.

### SETP

SETP is the “Simple Event Transfer Protocol”. SETP allows reliable delivery of events between SETP supporting systems. EventReporter, WinSyslog, and WinSyslog support SETP. EventReporter works as SETP Client Only. As such, it can forward events generated and gathered by them to central or intermediary SETP servers. WinSyslog EnterPrise Edition works as SETP client and server, only. The WinSyslog can operate both as a SETP server and client and as such also as a relay. It plays a vital role in a complex, distributed environment.

SETP was developed for MonitorWare. It allows synchronous communication between SETP clients and servers. With SETP, an event can be forwarded exactly as it was on the original event generating system. For example, if a syslog message is received on a remote system, that exact syslog message can be forwarded via as many SETP relays as is configured. During that relaying, no information from the original message is altered or lost. As such, each of the relays as well as the final SETP server will see the original source address, time stamps and message.

Furthermore, SETP guarantees reliable delivery. It is based on TCP, so each of the SETP peers know exactly that the communication partner can successfully receive and process the message. SETP guarantees that new events are only forwarded after the previous ones were successfully received and processed. SETP also checks for on the wire errors. Due to its characteristics, SETP can successfully be used in barely or occasionally connected environments like radio connected systems.

The SETP design is influenced by many industry standard movements, most notably the BEEP protocol and XML. However, SETP is optimized to have a very lightweight footprint. As such, it can be implemented even in low powered devices with little overhead.

### SMTP

The “Simple Mail Transfer Protocol”. This is an Internet standard for sending email messages. Virtually all major email systems are either based on SMTP or at least offer gateways to SMTP capable systems.

SMTP is used for sending email. It cannot be used to pick up email messages. For this purpose, protocols like POP3 or IMAP4 are required.

SMTP is highly standardized. As such, a standard email client can work with all SMTP compliant servers. In the public Internet, almost all providers offer SMTP compliant mail servers for their customer’s use.

### SNMP

SNMP stands for Simple Network Management Protocol. A set of standards for communication with devices connected to a TCP/IP network, like routers, hubs and switches. A device is said to be SNMP compatible if it can be monitored and/or controlled using SNMP messages.

SNMP messages are known as PDU’s - Protocol Data Units. Devices that are SNMP compatible contain SNMP ‘agent’ software to receive, send, and act upon SNMP messages. Software for managing devices via SNMP are available for every kind of commonly used computer and are often bundled along with the device they are designed to manage. Some SNMP software is designed to handle a wide variety of devices.

### Syslog Facility

Syslog Facility is one information field associated with a syslog message. It is defined by the Syslog protocol. It is meant to provide a very rough clue from what part of a system the message originated from. Traditionally, under UNIX, there are facilities like KERN (the OS kernel itself), LPD (the line printer daemon), and so on. There are also the *LOCAL_0* to *LOCAL_7* facilities, which were traditionally reserved for administrator and application use.

However, with the wide adoption of the syslog protocol, the facility field contents has become a little less clear. Most syslog enabled devices nowadays allow configuring any value as the facility. So it is basically left to distinguish different classes of syslog messages.

The facility can be very helpful to define rules that split messages for example to different log files based on the facility level.

### TCP

A reliable IP transport protocol. TCP communication ensures that no packets are lost in transit. As such, it is most useful in low-bandwidth or unreliable environments. Examples are slow WANs or packet radio networks.

Here you find information about Performance Tests and Results

### UDP

A non-reliable IP transport protocol. It provides best effort delivery. Typically, in LAN environments UDP packets are never lost. However, in WAN scenarios or with heavily loaded LANs, UDP packets might be lost.

Here you find information about Performance Tests and Results

### UTC

UTC is the so-called “universal coordinated time”. UTC was formerly referred to as “GMT” (Greenwich Mean Time) and is the basis of the international time zone system. For example, New York, USA is 5 hours behind UTC. So if it is 12 noon in New York, the UTC time is 5pm.

The MonitorWare line of products often uses UTC. UTC has the fast advantage of providing one consistent time notation, even if devices are across multiple time zones. This is extremely valuable if a central location is to consolidate events from senders in multiple time zones.

Using UTC might not be appropriate if a whole system is contained within a single time zone. As such, most time parameters inside the MonitorWare line of products can be configured to work with local time instead of UTC.

## WinSyslog product Event ID reference

Use this appendix to look up Windows Application Event Log entries generated by WinSyslog. Each Event ID links to its meaning, immediate checks, exact procedures, and escalation evidence.

### Scope

This reference covers product service diagnostics emitted by current supported builds. It does not cover Event IDs collected from other Windows applications or IDs chosen by an administrator in the Write to Windows Event Log action.

### WinSyslog Event ID troubleshooting procedures

Use these procedures from the applicable Event ID page. Each procedure includes commands, expected results, failure branches, recovery verification, and evidence collection.

- **`Collect an Event ID and neighboring product events`** — Preserve the complete event and the product events immediately before and after it.
- **`Collect evidence for an escalation-only runtime event`** — Capture a bounded reproducible support package without unsafe generic repair.
- **`Diagnose an action backlog or disk queue`** — Identify why queued work is not draining while preserving data.
- **`Diagnose log rotation and retention`** — Verify trigger, names, handles, destination access, and retention.
- **`Export configuration and collect a bounded debug log`** — Create a text configuration export and time-bounded debug capture, then disable debugging.
- **`Interpret a Windows or Winsock error code`** — Translate a numeric code without losing its operation or subsystem context.
- **`Repair the permitted-senders configuration`** — Authorize an intended sender without weakening the sender restriction.
- **`Resolve a destination and test its TCP port`** — Verify DNS, selected address, routing, and TCP establishment.
- **`Test an ODBC connection in the product context`** — Verify ODBC provider, architecture, authentication, connectivity, and a minimal query.
- **`Test an OLE DB connection in the product context`** — Verify OLE DB provider, architecture, authentication, connectivity, and a minimal query.
- **`Validate configuration and reload it safely`** — Back up, inspect, correct, and test the exact invalid configuration object.
- **`Verify a monitored remote service`** — Confirm resolution, transport, protocol, credentials, expected response, and timing.
- **`Verify a program or Windows-service control action`** — Check target, arguments, working directory, account rights, and positive result.
- **`Verify a UDP path without assuming delivery`** — Confirm receiver binding, firewall policy, and positive receipt for a paced sample.
- **`Verify an SNMP trap sender and receiver path`** — Confirm transport, endpoint, SNMP version, security/community, OIDs, and receipt.
- **`Verify Event Log channel access and bookmark state`** — Confirm channel existence, enablement, account access, and collection position.
- **`Verify File Monitor source state and encoding`** — Check source path, sharing, encoding, line endings, and read position.
- **`Verify file paths, permissions, and free space`** — Check expansion, existence, ACLs, service-account context, and storage.
- **`Verify listener binding and Windows Firewall rules`** — Confirm effective address, port, transport, owning process, and inbound policy.
- **`Verify Microsoft Message Queuing availability and access`** — Confirm feature, service, queue path, transaction mode, and send rights.
- **`Verify product license and feature entitlement state`** — Confirm product, version, validity, edition, and required feature without exposing license data.
- **`Verify sender, receiver, and queued-message recovery`** — Prove end-to-end recovery and backlog drainage.
- **`Verify service state, dependencies, and service account`** — Confirm service state, start mode, dependencies, account, and SCM errors.
- **`Verify SMTP connectivity and mail delivery`** — Separate DNS, TCP, TLS, authentication, relay, recipient, and downstream delivery.
- **`Verify TLS certificates, private keys, and permitted peers`** — Identify CA, certificate, private-key, trust-chain, and permitted-peer failures without exposing private key material.

### Event ID index

| Event ID | Meaning | Severity | Component |
|---|---|---|---|
| `100` | The service was installed | Information | Windows service lifecycle |

| | | | |
|---|---|---|---|
| **101** | The service was removed | Information | Windows service lifecycle |
| **102** | The service could not be removed | Error | Windows service lifecycle |
| **103** | The service control handler could not be installed | Error | Windows service lifecycle |
| **104** | The service initialization process failed | Error | Windows service lifecycle |
| **105** | The service was started | Information | Windows service lifecycle |
| **106** | The service received an unsupported control request | Error | Windows service lifecycle |
| **108** | The service was stopped | Information | Windows service lifecycle |
| **118** | Product license status reminder | Information | Licensing |
| **125** | Freeware mode reminder | Information | Licensing |
| **900** | Invalid license key rejected | Error | Licensing |
| **901** | Blocked license key rejected | Error | Licensing |
| **11000** | SETP connection closed by the receiver | Warning | SETP sender |
| **11001** | SETP TLS session closed | Warning | SETP sender |
| **11002** | SETP delivery delayed in the action cache | Warning | SETP sender |
| **11003** | SETP delivery failed | Error | SETP sender |
| **11004** | SETP session restored | Information | SETP sender |
| **11005** | Configured feature is not licensed | Warning | Licensing |
| **11006** | Call Ruleset action could not find its target ruleset | Error | Call Ruleset action |
| **11007** | Action configuration: required timer could not be created | Warning | Action configuration |
| **11008** | Action configuration: configured action feature is unavailable | Warning | Action configuration |
| **11009** | A queued action message could not be delivered | Warning | Action disk queue |
| **11010** | ODBC database action: database operation failed | Error | ODBC database action |
| **11011** | ODBC database action: database operation failed | Error | ODBC database action |
| **11012** | OLE DB action: database operation failed | Error | OLE DB action |
| **11013** | OLE DB action: OLE DB operation failed | Error | OLE DB action |
| **11014** | Action processing: action initialization failed | Error | Action processing |
| **11015** | Microsoft Message Queuing action: Microsoft Message Queuing operation failed | Error | Microsoft Message Queuing action |
| **11016** | An SNMP trap variable contains an invalid OID | Warning | Send SNMP Trap action |
| **11017** | Forward Syslog action: action initialization failed | Error | Forward Syslog action |
| **11018** | A queued Forward Syslog message could not be delivered | Warning | Forward Syslog disk queue |
| **11019** | Syslog Queue action has no queue name | Error | Syslog Queue action |
| **11020** | Control Windows Service action could not open Service Control Manager | Error | Control Windows Service action |
| **11021** | Action processing: configured action feature is unavailable | Warning | Action processing |
| **11022** | Action processing: action type is missing from configuration | Error | Action processing |
| **11023** | Action processing: invalid action type in configuration | Error | Action processing |
| **11024** | Configured action type is not recognized | Error | Rule engine action |
| **11025** | Action processing: invalid action type in configuration | Error | Action processing |
| **11026** | Configured action failed after its retries | Error | Rule engine action |

| | | | |
|---|---|---|---|
| **11027** | Configured action raised an unknown exception at its retry limit | Error | Rule engine action |
| **11028** | Configured action raised an unknown exception below its retry limit | Error | Rule engine action |
| **11029** | ODBC database action: database operation failed | Error | ODBC database action |
| **11030** | ODBC database action: database operation failed | Error | ODBC database action |
| **11031** | OLE DB action: database operation failed | Error | OLE DB action |
| **11032** | OLE DB action: OLE DB operation failed | Error | OLE DB action |
| **11033** | Post Process action ignored an unsupported parse type | Error | Post Process action |
| **11034** | SETP sender: TLS client initialization failed | Error | SETP sender |
| **11035** | SETP sender: SETP session initialization failed | Error | SETP sender |
| **11036** | RELP delivery failed and the message was queued | Warning | Send RELP action |
| **11037** | Send to Communications Port action could not write a byte | Error | Send to Communications Port action |
| **11038** | SNMP trap transmission failed | Error | Send SNMP Trap action |
| **11039** | Forward Syslog delivery failed and the message was queued | Warning | Forward Syslog action |
| **11040** | Forward Syslog delivery failed without queue protection | Error | Forward Syslog action |
| **11041** | Start Program action received an unexpected wait result | Error | Start Program action |
| **11042** | Write to File action has an invalid output format | Error | Write to File action |
| **11043** | A sender connection was refused by the permitted-senders limit | Error | Client connection licensing |
| **11044** | A sender connection exceeded the licensed client limit | Error | Client connection licensing |
| **11045** | File-based configuration: file-based configuration could not be loaded | Error | File-based configuration |
| **11046** | File-based configuration: file-based configuration could not be loaded | Error | File-based configuration |
| **11047** | Configuration: invalid host or IP address | Error | Configuration |
| **11048** | Network and TLS transport: unknown runtime error | Warning | Network and TLS transport |
| **11049** | TLS CA certificate file could not be loaded | Warning | TLS certificate loading |
| **11050** | TLS certificate or private key could not be loaded | Warning | TLS certificate loading |
| **11051** | TLS client certificate verification failed | Warning | TLS client certificate verification |
| **11052** | TLS client certificate identity is unavailable | Warning | TLS client certificate verification |
| **11053** | TLS client certificate does not match a permitted peer | Warning | TLS peer authorization |
| **11054** | TLS certificate-name authentication has no permitted peers | Warning | TLS peer authorization |
| **11055** | TLS client certificate fingerprint could not be computed | Warning | TLS peer authorization |
| **11056** | TLS certificate-fingerprint authentication has no permitted peers | Warning | TLS peer authorization |
| **11057** | Network and TLS transport: TLS server initialization failed | Warning | Network and TLS transport |
| **11058** | Filter engine: regular expression initialization failed | Error | Filter engine |
| **11059** | Monitoring service: service component initialization failed | Error | Monitoring service |
| **11060** | MonitorWare Echo service: MonitorWare Echo operation raised an exception | Warning | MonitorWare Echo service |

| | | | |
|---|---|---|---|
| **11061** | MonitorWare Echo service: MonitorWare Echo operation raised an exception | Error | MonitorWare Echo service |
| **11062** | MonitorWare Echo service: MonitorWare Echo operation raised an exception | Error | MonitorWare Echo service |
| **11063** | MonitorWare Echo service: MonitorWare Echo operation raised an unknown exception | Error | MonitorWare Echo service |
| **11064** | CPU Monitor service: CPU monitor raised an exception | Error | CPU Monitor service |
| **11065** | CPU Monitor service: CPU monitor raised an unknown exception | Error | CPU Monitor service |
| **11066** | Database Monitor service: database operation failed | Error | Database Monitor service |
| **11067** | Database Monitor service: database operation failed | Error | Database Monitor service |
| **11068** | Database Monitor service: database monitor ID field warning | Warning | Database Monitor service |
| **11069** | Database Monitor service: database operation failed | Error | Database Monitor service |
| **11070** | Database Monitor service: database operation failed | Error | Database Monitor service |
| **11071** | Disk Space Monitor could not complete a disk-space check | Error | Disk Space Monitor service |
| **11072** | DTLS listener: listener could not bind to its configured address | Error | DTLS listener |
| **11073** | DTLS listener stopped after an unknown exception | Error | DTLS listener |
| **11074** | File Monitor service: file monitoring warning | Warning | File Monitor service |
| **11075** | File Monitor service: configured file could not be opened | Error | File Monitor service |
| **11076** | File Monitor service: configured file could not be opened | Error | File Monitor service |
| **11077** | File Monitor service: file monitoring operation failed | Error | File Monitor service |
| **11078** | File Monitor service: file monitoring operation failed | Error | File Monitor service |
| **11079** | File Monitor service: configured file could not be opened | Error | File Monitor service |
| **11080** | File Monitor service: configured file could not be opened | Error | File Monitor service |
| **11081** | File Monitor service: file monitoring operation failed | Error | File Monitor service |
| **11082** | FTP Probe service: probe operation raised an exception | Error | FTP Probe service |
| **11083** | FTP Probe service: probe operation raised an unknown exception | Error | FTP Probe service |
| **11084** | Heartbeat service: heartbeat operation raised an exception | Error | Heartbeat service |
| **11085** | Heartbeat service: heartbeat operation raised an unknown exception | Error | Heartbeat service |
| **11086** | HTTP Probe service: probe operation raised an exception | Error | HTTP Probe service |
| **11087** | HTTP Probe service: probe operation raised an unknown exception | Error | HTTP Probe service |
| **11088** | IMAP Probe service: probe operation raised an exception | Error | IMAP Probe service |
| **11089** | IMAP Probe service: probe operation raised an unknown exception | Error | IMAP Probe service |
| **11090** | SETP listener stopped after a protocol server error | Error | SETP listener |

| | | | |
|---|---|---|---|
| **11091** | SETP receiver: SETP operation raised an exception | Warning | SETP receiver |
| **11092** | SETP receiver: SETP operation raised an exception | Error | SETP receiver |
| **11093** | SETP receiver: SETP operation raised an exception | Error | SETP receiver |
| **11094** | SETP receiver: SETP operation raised an unknown exception | Error | SETP receiver |
| **11095** | NNTP Probe service: probe operation raised an exception | Error | NNTP Probe service |
| **11096** | NNTP Probe service: probe operation raised an unknown exception | Error | NNTP Probe service |
| **11097** | Event Log Monitor service: Windows Event Log operation failed | Error | Event Log Monitor service |
| **11098** | Event Log Monitor service: Windows Event Log could not be opened | Error | Event Log Monitor service |
| **11099** | Event Log Monitor service: unknown runtime error | Error | Event Log Monitor service |
| **11100** | A corrupted Windows Event Log record was skipped | Warning | Event Log Monitor service |
| **11101** | Event reporter service | Error | Event Log Monitor service |
| **11102** | Event Log Monitor service: Windows Event Log record was unreadable | Error | Event Log Monitor service |
| **11103** | Event Log Monitor service: Windows Event Log clear target already exists | Error | Event Log Monitor service |
| **11104** | Event Log Monitor service: Windows Event Log could not be cleared | Error | Event Log Monitor service |
| **11105** | Event Log Monitor service: Windows Event Log clear retry succeeded | Information | Event Log Monitor service |
| **11106** | Windows Event Log cleared and saved | Information | Event Log Monitor service |
| **11107** | Windows Event Log cleared | Information | Event Log Monitor service |
| **11108** | Event Log Monitor service: Windows Event Log operation failed | Error | Event Log Monitor service |
| **11109** | Event Log Monitor service: Windows Event Log operation failed | Error | Event Log Monitor service |
| **11110** | Event Log Monitor service: licensed Event Log Monitor limit reached | Error | Event Log Monitor service |
| **11111** | Windows Service Monitor: Windows Service Monitor raised an exception | Error | Windows Service Monitor |
| **11112** | Syslog listener: service component initialization failed | Error | Syslog listener |
| **11113** | Syslog listener: service component initialization failed | Error | Syslog listener |
| **11114** | Syslog listener: listener could not bind to its configured address | Warning | Syslog listener |
| **11115** | Syslog listener: listener could not bind to its configured address | Error | Syslog listener |
| **11116** | Passive Syslog listener stopped after an unknown exception | Error | Passive Syslog listener |
| **11117** | Ping Probe service stopped after a reported runtime error | Error | Ping Probe service |
| **11118** | Ping Probe service stopped after an unknown exception | Error | Ping Probe service |
| **11119** | POP3 Probe service: probe operation raised an exception | Error | POP3 Probe service |

| | | | |
|---|---|---|---|
| **11120** | POP3 Probe service: probe operation raised an unknown exception | Error | POP3 Probe service |
| **11121** | RELP listener: listener could not bind to its configured address | Error | RELP listener |
| **11122** | RELP listener stopped after an unknown exception | Error | RELP listener |
| **11123** | Serial Port Monitor: serial port operation failed | Error | Serial Port Monitor |
| **11124** | Serial Port Monitor: serial port operation raised an exception | Error | Serial Port Monitor |
| **11125** | Serial Port Monitor: serial port operation raised an unknown exception | Error | Serial Port Monitor |
| **11126** | SMTP listener: SMTP listener raised an exception | Warning | SMTP listener |
| **11127** | SMTP listener: SMTP listener raised an exception | Error | SMTP listener |
| **11128** | SMTP listener: SMTP listener raised an exception | Error | SMTP listener |
| **11129** | SMTP listener: SMTP listener raised an unknown exception | Error | SMTP listener |
| **11130** | SMTP Probe service: probe operation raised an exception | Error | SMTP Probe service |
| **11131** | SMTP Probe service: probe operation raised an unknown exception | Error | SMTP Probe service |
| **11132** | SNMP Monitor service: SNMP operation raised an exception | Warning | SNMP Monitor service |
| **11133** | SNMP Monitor service: SNMP operation raised an exception | Error | SNMP Monitor service |
| **11134** | SNMP Monitor service: SNMP operation raised an unknown exception | Error | SNMP Monitor service |
| **11135** | SNMP Monitor service: SNMP operation raised an exception | Error | SNMP Monitor service |
| **11136** | SNMP Monitor service: SNMP operation raised an unknown exception | Error | SNMP Monitor service |
| **11137** | SNMP Trap Receiver: SNMP operation raised an exception | Warning | SNMP Trap Receiver |
| **11138** | SNMP Trap Receiver: SNMP operation raised an exception | Error | SNMP Trap Receiver |
| **11139** | SNMP Trap Receiver: SNMP operation raised an exception | Error | SNMP Trap Receiver |
| **11140** | SNMP Trap Receiver: SNMP operation raised an unknown exception | Error | SNMP Trap Receiver |
| **11141** | Syslog listener: BEEP listener option could not be applied | Error | Syslog listener |
| **11142** | Syslog listener: BEEP listener initialization failed | Error | Syslog listener |
| **11143** | Syslog listener: BEEP listener port could not be applied | Error | Syslog listener |
| **11144** | Syslog listener: listener could not bind to its configured address | Warning | Syslog listener |
| **11145** | Syslog listener: listener could not bind to its configured address | Error | Syslog listener |
| **11146** | Syslog listener stopped after an unknown exception | Error | Syslog listener |
| **11147** | Event Log Monitor V2 service: required timer could not be created | Warning | Event Log Monitor V2 service |
| **11148** | Event Log Monitor could not create its saved bookmark | Error | Event Log Monitor V2 service |
| **11149** | Event Log Monitor V2 service: general runtime error | Warning | Event Log Monitor V2 service |

| | | | |
|---|---|---|---|
| **11150** | Event Log Monitor V2 service: Windows Event Log operation failed | Error | Event Log Monitor V2 service |
| **11151** | Event Log Monitor V2 service: Windows Event Log operation failed | Error | Event Log Monitor V2 service |
| **11152** | A queued event has an unsupported information-unit type | Error | Event reconstruction |
| **11153** | Existing archive file was renamed before log rotation | Warning | Log rotation |
| **11154** | Log rotation was deferred because an archive destination could not be renamed | Warning | Log rotation |
| **11155** | Cross-volume archive copy succeeded but source cleanup failed | Warning | Log rotation |
| **11156** | Rotated log could not be moved to its archive destination | Warning | Log rotation |
| **11157** | Existing archive could not be renamed to prevent overwrite | Warning | Log rotation |
| **11158** | Empty log-rotation queue file could not be deleted | Warning | Log rotation queue |
| **11159** | Corrupt log-rotation queue file could not be quarantined | Warning | Log rotation queue |
| **11160** | Corrupt log-rotation queue file was quarantined | Warning | Log rotation queue |
| **11161** | Configuration reload kept the existing log-rotation workers | Warning | Log rotation |
| **11162** | Log-rotation jobs cannot be persisted because the data directory is unset | Warning | Log rotation queue |
| **11163** | Persisted log-rotation queue file could not be read | Warning | Log rotation queue |
| **11164** | Email action: invalid SMTP recipient address | Error | Email action |
| **11165** | Email action: SMTP recipient address is missing | Error | Email action |
| **11166** | OLE DB action: OLE DB operation failed | Error | OLE DB action |
| **11167** | Runtime resource management: memory reserve is in use | Warning | Runtime resource management |
| **11168** | Runtime resource management: memory allocation failed | Error | Runtime resource management |
| **11169** | Individual Ping Probe stopped after an unknown exception | Error | Ping Probe |
| **11170** | Queue worker received an unexpected wake-up | Error | Main message queue |
| **11171** | Queue manager: unknown runtime error | Error | Queue manager |
| **11172** | Queue manager: unknown runtime error | Error | Queue manager |
| **11173** | An event failed while the main queue applied its rules | Error | Main message queue |
| **11174** | Queue manager: queue capacity condition | Information | Queue manager |
| **11175** | Queue manager: queue capacity condition | Warning | Queue manager |
| **11177** | Queue manager: queue capacity condition | Error | Queue manager |
| **11178** | Queue manager: disk queue cache operation failed | Error | Queue manager |
| **11179** | Queue manager: disk queue cache operation failed | Error | Queue manager |
| **11180** | Queue manager: disk queue cache operation failed | Error | Queue manager |
| **11181** | Queue manager: disk queue cache operation failed | Error | Queue manager |
| **11182** | Queue manager: disk queue cache operation failed | Error | Queue manager |
| **11183** | Configuration reload was skipped because the log-rotation scheduler did not stop | Warning | Configuration reload |
| **11184** | Service configuration: memory reserve could not be allocated | Warning | Service configuration |
| **11185** | No product input service is configured | Error | Service configuration |

| | | | |
|---|---|---|---|
| **11186** | The trial expired and the service was disabled | Error | Licensing |
| **11187** | A configured input service type is not recognized | Error | Service configuration |
| **11188** | Service configuration: configured service feature is unavailable | Warning | Service configuration |
| **11189** | Input service could not find its assigned ruleset | Error | Service configuration |
| **11190** | Service configuration: configured feature is unavailable | Warning | Service configuration |
| **11191** | Service configuration: log rotation is not licensed | Information | Service configuration |
| **11192** | No configured product input service could be started | Error | Service startup |
| **11193** | Service configuration: service configuration raised an unknown exception | Warning | Service configuration |
| **11194** | Product runtime: message compression failed | Warning | Product runtime |
| **11195** | Disk queue: runtime data error | Error | Disk queue |
| **11196** | Disk queue: runtime data error | Error | Disk queue |
| **11197** | Disk queue: runtime data error | Error | Disk queue |
| **11198** | TCP and TLS listener: TCP listener fatal error | Error | TCP and TLS listener |
| **11199** | TCP and TLS listener: TLS server initialization failed | Warning | TCP and TLS listener |
| **11200** | TCP and TLS listener: TCP listener fatal error | Error | TCP and TLS listener |
| **11201** | TCP and TLS listener: TCP listener fatal error | Error | TCP and TLS listener |
| **11202** | TCP listener could not assign an accepted socket | Error | TCP and TLS listener |
| **11210** | Product runtime: configured advanced feature is unavailable | Error | Product runtime |
| **11212** | System exception while compiling an NT Event Log expression | Error | Event Log Monitor service |
| **11213** | System exception while compiling an NT Event Log category | Error | Event Log Monitor service |
| **11214** | System exception in the queue worker thread | Error | Queue manager |
| **11215** | A debug error was forwarded to the Windows Event Log | Warning | Diagnostic forwarding |
| **11216** | Control Windows Service action could not open the target service | Error | Control Windows Service action |
| **11217** | Control Windows Service action could not start the target service | Error | Control Windows Service action |
| **11218** | Control Windows Service action could not stop the target service | Error | Control Windows Service action |
| **11219** | Control Windows Service action could not pause the target service | Error | Control Windows Service action |
| **11220** | Control Windows Service action could not continue the target service | Error | Control Windows Service action |
| **11221** | Target service did not reach the requested state before timeout | Error | Control Windows Service action |
| **11222** | Consumed log-rotation queue file could not be deleted | Warning | Log rotation queue |
| **11223** | Protected service shutdown deadline expired | Error | Service shutdown |
| **11224** | Configuration reload shutdown incomplete | Error | Service shutdown |
| **11225** | RELP listener shutdown wake fallback | Warning | RELP listener |
| **11226** | SNMP listener shutdown wake fallback | Warning | SNMP trap listener |
| **11227** | Event rejected after shutdown intake closed | Warning | Main message queue |
| **11228** | Action disk queue flush failed during shutdown | Error | Action persistence |

| | | | |
|---|---|---|---|
| **11229** | Message Ringbuffer action is not registered | Error | Message Ringbuffer action |

## Known issues and errata

### Known issues and errata

This section documents confirmed product behavior that can affect operation or the interpretation of collected data. Each notice identifies the affected products and service versions, the practical impact, available workarounds, and the planned or released correction.

#### Current errata

- ERR-2026-001: Incorrect source address for received SNMP traps
- ERR-2026-002: WinSyslog update reports success but program files are not upgraded correctly

## External reference

- Version History
- WinSyslog 26.07 release notes

# Copyrights

This documentation as well as the actual WinSyslog product is copyrighted by Adiscon GmbH, Germany. To learn more about other Adiscon products, please visit https://www.adiscon.com/en/products.

We acknowledge using these following third party tools. Here are the download links:

**Openssl-3.2.1:** https://www.openssl.org/source/openssl-3.2.1.tar.gz **Liblogging 0.7.1:** https://github.com/Rsyslog/liblogging/archive/refs/tags/v0.7.1.tar.gz **Librelp 1.11.0:** https://github.com/Rsyslog/librelp/archive/refs/tags/v1.10.0.tar.gz **Libfastjson-0.99.8:** https://github.com/Rsyslog/libfastjson/archive/refs/tags/v0.99.8.tar.gz

**Liblognorm 0.3.5** https://github.com/Rsyslog/liblognorm/archive/refs/tags/v0.3.5.tar.gz

Microsoft, Windows, and the Windows logo are trademarks, or registered trademarks of Microsoft Corporation in the United States and/or other countries.

Other mentioned trademarks are for reference only. They belong to their respective owners.

- **genindex**

# Index


### A

Accessing Properties

### C

Command Line Switches
Comparison of Properties
Complex Filter Conditions
Connect to Computer
Customer Properties

### D

Database Monitor

### E

engine only install
Event Properties
Event-Specific Properties

### F

FromPos
FTP

### H

HTTP

### I

IETF
IMAP
Information Units
Installation
IPv6

### M

Millisecond

### N

NNTP

### O

Options

### P

POP3
Property

### R

Registry File
Registry Paths
RELP
Resource ID
RFC 3164
RFC 3195
RFC 5424
Rule Engine
Rules

### S

SETP
SMTP
SNMP
Standard Properties
Syslog Facility
Syslog Message Properties
System Properties

### T

TCP
ToPos

### U

UDP
UTC

### W

Windows Event Log Properties
Windows Event Log V2 Properties